

Bartłomiej Beliczynski
Andrzej Dzieliński
Marcin Iwanowski
Bernardete Ribeiro (Eds.)

LNCS 4431

Adaptive and Natural Computing Algorithms

8th International Conference, ICANNGA 2007
Warsaw, Poland, April 2007
Proceedings, Part I

1
Part I

 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

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

University of Dortmund, Germany

Madhu Sudan

Massachusetts Institute of Technology, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Moshe Y. Vardi

Rice University, Houston, TX, USA

Gerhard Weikum

Max-Planck Institute of Computer Science, Saarbruecken, Germany

Bartłomiej Beliczynski Andrzej Dzielinski
Marcin Iwanowski Bernardete Ribeiro (Eds.)

Adaptive and Natural Computing Algorithms

8th International Conference, ICANNGA 2007
Warsaw, Poland, April 11-14, 2007
Proceedings, Part I

Volume Editors

Bartłomiej Beliczynski

Andrzej Dzielinski

Marcin Iwanowski

Warsaw University of Technology

Institute of Control and Industrial Electronics

ul. Koszykowa 75, 00-662 Warszawa, Poland

E-mail: {B.Beliczynski,A.Dzielinski,M.Iwanowski}@ee.pw.edu.pl

Bernardete Ribeiro

University of Coimbra

Department of Informatics Engineering

Polo II, 3030-290 Coimbra, Portugal

E-mail: bribeiro@dei.uc.pt

Library of Congress Control Number: 2007923870

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

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

ISSN 0302-9743

ISBN-10 3-540-71589-4 Springer Berlin Heidelberg New York

ISBN-13 978-3-540-71589-4 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 is a part of Springer Science+Business Media

springer.com

© Springer-Verlag Berlin Heidelberg 2007

Printed in Germany

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

Printed on acid-free paper SPIN: 12041114 06/3180 5 4 3 2 1 0

Preface

The ICANNGA series of conferences has been organized since 1993 and has a long history of promoting the principles and understanding of computational intelligence paradigms within the scientific community. Starting in Innsbruck, in Austria (1993), then Ales in France (1995), Norwich in England (1997), Portoroz in Slovenia (1999), Prague in Czech Republic (2001), Roanne in France (2003) and finally Coimbra in Portugal (2005), the ICANNGA series has established itself as a reference for scientists and practitioners in this area. The series has also been of value to young researchers wishing both to extend their knowledge and experience and to meet experienced professionals in their fields.

In a rapidly advancing world, where technology and engineering change dramatically, new challenges in computer science compel us to broaden the conference scope in order to take into account new developments. Nevertheless, we have kept the acronym ICANNGA which, since the Coimbra conference in 2005, stands for International Conference on Adaptive and Natural Computing Algorithms.

The 2007 conference, the eighth in the ICANNGA series, took place at the Warsaw University of Technology in Poland, drawing on the experience of previous events and following the same general model, combining technical sessions, including plenary lectures by renowned scientists, with tutorials and workshop panels.

The Warsaw edition of ICANNGA attracted many scientists from all over the world. We received 474 mostly high-quality submissions from 40 countries. After rigorous review involving more than 160 experts in their fields, 178 papers were accepted and included in the proceedings. The acceptance rate was only 38%, enforcing a high standard of papers. The conference proceedings are published in two volumes of Springer's *Lecture Notes in Computer Science*.

The first volume of the proceedings is primarily concerned with issues related to various concepts and methods of optimization, evolutionary computations, genetic algorithms, particle swarm optimization, fuzzy and rough systems. Additionally there is also a set of papers devoted to clustering and classification. The second volume is mainly concerned with neural networks theory and applications, support vector machines, biomedical and biometrics applications, computer vision, control and robotics.

ICANNGA 2007 enjoyed plenary lectures presented by distinguished scientists: Shun-ichi Amari from Japan, Ryszard Tadeusiewicz and Janusz Kacprzyk from Poland, Kevin Warwick and Rafal Zbikowski from England.

We would like to thank the International Advisory Committee for their guidance, advice and discussions. Our special gratitude is devoted to the Program Committee and reviewers. They have done a wonderful job of shaping the conference image.

Camera-ready version of the papers were carefully examined and verified by Wiktor Malesza, Konrad Markowski, Tomasz Toczyski and Maciej Twardy. A number of people from our Electrical Engineering Faculty, the Control Division Staff members and the PhD students were involved in various conference tasks, supporting the conference secretariat and maintaining multimedia equipment. We greatly appreciate all they have done.

We also wish to thank our publisher, especially Alfred Hofmann the Editor-in-Chief of LNCS and Anna Kramer for their support and collaboration.

Finally, the conference was made up of papers and presentations prepared by our contributors and participants. Most of our gratitude is directed to them.

April 2007

Bartłomiej Beliczynski
Andrzej Dzielinski
Marcin Iwanowski
Bernardete Ribeiro

Organization

Advisory Committee

Rudolf Albrecht, University of Innsbruck, Austria
Andrej Dobnikar, University of Ljubljana, Slovenia
Vera Kurkova, Academy of Sciences of the Czech Republic, Czech Republic
David Pearson, University Jean Monnet, France
Bernardete Ribeiro, University of Coimbra, Portugal
Nigel Steele, Coventry University, UK

Program Committee

| | |
|---|------------------------------|
| Bartłomiej Beliczynski, Poland (Chair) | Vera Kurkova, Czech Republic |
| Rudolf Albrecht, Austria | Pedro Larranaga, Spain |
| Gabriela Andrejkova, Slovakia | Francesco Masulli, Italy |
| Paulo de Carvalho, Portugal | Leila Mokhnache, Algeria |
| Ernesto Costa, Portugal | Roman Neruda, Czech Republic |
| Andrej Dobnikar, Slovenia | Stanislaw Osowski, Poland |
| Marco Dorigo, Belgium | Nikola Pavesic, Slovenia |
| Antonio Dourado, Portugal | David Pearson, France |
| Gerard Dray, France | Maria Pietrzak-David, France |
| Andrzej Dzielinski, Poland | Colin Reeves, UK |
| Jorge Henriques, Portugal, | Bernardete Ribeiro, Portugal |
| Katerina Hlavackova-Schindler, Austria | Henrik Saxen, Finland |
| Osamu Hoshino, Japan | Marcello Sanguineti, Italy |
| Janusz Kacprzyk, Poland | Jiri Sima, Czech Republic |
| Tadeusz Kaczorek, Poland | Catarina Silva, Portugal |
| Paul C. Kainen, USA | Nigel Steele, UK |
| Helen Karatza, Greece | Miroslaw Swiercz, Poland |
| Miroslav Karny, Czech Republic | Ryszard Tadeusiewicz, Poland |
| Marian P.Kazmierkowski Poland | Tatiana Tambouratzis, Greece |
| Mario Koeppen, Germany | Kevin Warwick, UK |
| Jozef Korbicz, Poland | Stanislaw H. Zak, USA |

Organizing Committee

Bartłomiej Beliczynski (Chair)
Bernardete Ribeiro (Past Chair)
Witold Czajewski (Technical Support, Conference Events)
Andrzej Dzielinski (Reviewing Process)
Waldemar Graniszewski (Social Program)
Marcin Iwanowski (Conference Coordinator; Proceedings, WWW)
Grazyna Rabij (Finances)

Reviewers

| | |
|------------------------|-------------------------------|
| Rudolf Albrecht | Soowhan Han |
| Krzysztof Amborski | Zenon Hendzel |
| Gabriela Andrejkova | Jorge Henriques |
| Jaroslav Arabas | Mika Hirvensalo |
| Piotr Arabas | Katarina Hlavackova-Schindler |
| Prasanna Balaprakash | Osamu Hoshino |
| Bartłomiej Beliczynski | Yanhai Hu |
| Conrad Bielski | Ben Hutt |
| Fatih Mehmet Botsali | Naohiro Ishii |
| Cyril Brom | Marcin Iwanowski |
| Pawel Buczynski | Wojciech Jedruch |
| Paulo de Carvalho | Tatiana Jaworska |
| Hasan Huseyin Celik | Piotr Jedrzejowicz |
| Leszek Chmielewski | Sangbae Jeong |
| YoungSik Choi | Marcel Jirina |
| Michal Choras | Tomasz Kacprzak |
| Ryszard Choras | Janusz Kacprzyk |
| Gyo-Bum Chung | Tadeusz Kaczorek |
| Andrzej Cichocek | Paul C. Kainen |
| Ernesto Costa | Helen Karatza |
| David Coufal | Andrzej Karbowski |
| Boguslaw Cyganek | Ali Karci |
| Witold Czajewski | Miroslav Karny |
| Włodzimierz Dabrowski | Włodzimierz Kasprzak |
| Dariusz Krol | Marian P. Kazmierkowski |
| Guy De Tre | Adnan Khashman |
| Andrej Dobnikar | Chang-Soo Kim |
| Antonio Dourado | Il-Hwan Kim |
| Gerard Dray | Kwang-Baek Kim |
| Andrzej Dzielinski | Mi-Young Kim |
| Mehmet Onder Efe | Mario Koeppen |
| Maria Ganzha | Jozef Korbicz |
| Waldemar Graniszewski | Anna Korzynska |

Jacek Kozak
Wojciech Kozinski
Marek Kowal
Petra Kudova
Piotr Kulczycki
Vera Kurkova
Halina Kwasnicka
Bogdan Kwolek
Pedro Larranaga
Inbok Lee
Kidong Lee
Jun-Seok Lim
Hong-Dar Lin
Rafal Lopatka
Jacek Mandziuk
Mariusz Mlynarczuk
Mariusz Malinowski
Marcin Mrugalski
Konrad Markowski
Francesco Masulli
Yuri Merkuryev
Zbigniew Mikrut
Leila Mokhanche
Marco Montes de Oca
Jose Moreno
Nadia Nedjah
Roman Neruda
Mariusz Nieniewski
Joanna Nowak
Piotr Nowak
Marek Ogiela
Wlodzimierz Ogryczak
Stanislaw Osowski
Andrzej Pacut
Henryk Palus
Marcin Paprzycki
Byung Joo Park
JungYong Park
Kiejin Park
Miroslaw Parol
Krzysztof Patan
Nikola Pavesic
David W. Pearson
Daniel Prusa
Artur Przelaskowski

Jochen Radmer
Remigiusz Rak
Sarunas Raudys
Kiril Ribarov
Bernardete Ribeiro
Martin Rimnac
Claudio M. Rocco S.
Miguel Rocha
Przemyslaw Rokita
Maciej Romaniuk
Maciej Slawinski
Stanislav Saic
Marcello Sanguineti
José Santos Reyes
Henrik Saxen
Franciszek Seredynski
Dongmin Shin
Barbara Siemiatkowska
Dominik Sierociuk
Catarina Silva
Jiri Sima
Slawomir Skoneczny
Andrzej Sluzek
Czeslaw Smutnicki
Pierre Soille
Oleksandr Sokolov
Nigel Steele
Barbara Strug
Pawel Strumillo
Bartlomiej Sulikowski
Miroslaw Swiercz
Krzysztof Szczypiorski
Jarosaw Szostakowski
Wojciech Szynekiewicz
Ryszard Tadeusiewicz
Tatiana Tambouratzis
Jorge Tavares
Tomasz Toczyski
Krzysztof Trojanowski
George A. Tsihrintzis
Pavel Vacha
Armando Vieira
Wen-Pai Wang
Slawomir Wierzchon
Anna Wilbik

Marcin Witczak
Maciej Wygralak
Mykhaylo Yatsymirskyy
Slawomir Zadrozny

Cezary Zielinski
Stanislaw H. Zak

Organizers

ICANNGA 2007 was organized by the Control Division of the Institute of Control and Industrial Electronics, Faculty of Electrical Engineering, Warsaw University of Technology, Poland.

Table of Contents – Part I

Evolutionary Computation

| | |
|---|-----|
| Evolutionary Induction of Decision Trees for Misclassification Cost Minimization | 1 |
| <i>Marek Krętownski and Marek Grześ</i> | |
| DNA Based Evolutionary Approach for Microprocessor Design Automation | 11 |
| <i>Nagarajan Venkateswaran, Arjun Kumeresh, and Harish Chandran</i> | |
| Multiple Sequence Alignment with Evolutionary-Progressive Method . . . | 23 |
| <i>Paweł Kupis and Jacek Mańdziuk</i> | |
| Optimal Design Centring Through a Hybrid Approach Based on Evolutionary Algorithms and Monte Carlo Simulation | 31 |
| <i>Luis Pierluissi and Claudio M. Rocco S.</i> | |
| A New Self-adaptative Crossover Operator for Real-Coded Evolutionary Algorithms | 39 |
| <i>Manuel E. Gegúndez, Pablo Palacios, and José L. Álvarez</i> | |
| Wavelet Enhanced Analytical and Evolutionary Approaches to Time Series Forecasting | 49 |
| <i>Bartosz Kozłowski</i> | |
| Gradient Based Stochastic Mutation Operators in Evolutionary Multi-objective Optimization | 58 |
| <i>Pradyumn Kumar Shukla</i> | |
| Co-evolutionary Multi-agent System with Predator-Prey Mechanism for Multi-objective Optimization | 67 |
| <i>Rafał Dreżewski and Leszek Siwik</i> | |
| Optical Design with Epsilon-Dominated Multi-objective Evolutionary Algorithm | 77 |
| <i>Shaine Joseph, Hyung W. Kang, and Uday K. Chakraborty</i> | |
| Boosting the Performance of a Multiobjective Algorithm to Design RBFNNs Through Parallelization | 85 |
| <i>Alberto Guillén, Ignacio Rojas, Jesus González, Hector Pomares, Luis J. Herrera, and Ben Paechter</i> | |
| Immune Algorithm Versus Differential Evolution: A Comparative Case Study Using High Dimensional Function Optimization | 93 |
| <i>Vincenzo Cutello, Natalio Krasnogor, Giuseppe Nicosia, and Mario Pavone</i> | |
| Self-adaptive Evolutionary Methods in Designing Skeletal Structures . . . | 102 |
| <i>Adam Borkowski and Piotr Nikodem</i> | |

| | |
|---|-----|
| An Evolutionary Approach to Task Graph Scheduling | 110 |
| <i>Saeed Parsa, Shahriar Lotfi, and Naser Lotfi</i> | |
| Universal Quantum Gates Via Yang-Baxterization of Dihedral Quantum Double | 120 |
| <i>Mario Vélez and Juan Ospina</i> | |
| Evolutionary Bi-objective Learning with Lowest Complexity in Neural Networks: Empirical Comparisons | 128 |
| <i>Yamina Mohamed Ben Ali</i> | |
| Improving the Quality of the Pareto Frontier Approximation Obtained by Semi-elitist Evolutionary Multi-agent System Using Distributed and Decentralized Frontier Crowding Mechanism | 138 |
| <i>Leszek Siwik and Marek Kisiel-Dorohinicki</i> | |
| On Semantic Properties of Interestingness Measures for Extracting Rules from Data | 148 |
| <i>Mondher Maddouri and Jamil Gammoudi</i> | |
| Genetic Algorithms | |
| A New Mutation Operator for the Elitism-Based Compact Genetic Algorithm | 159 |
| <i>Rafael R. Silva, Heitor S. Lopes, and Carlos R. Erig Lima</i> | |
| Genetic Programming for Proactive Aggregation Protocols | 167 |
| <i>Thomas Weise, Kurt Geihs, and Philipp A. Baer</i> | |
| Automatic Synthesis for Quantum Circuits Using Genetic Algorithms | 174 |
| <i>Cristian Ruican, Mihai Udrescu, Lucian Prodan, and Mircea Vladutiu</i> | |
| Clonal Selection Approach with Mutations Based on Symmetric α -Stable Distributions for Non-stationary Optimization Tasks | 184 |
| <i>Krzysztof Trojanowski</i> | |
| Minimizing Cycle Time of the Flow Line – Genetic Approach with Gene Expression | 194 |
| <i>Paweł Dąbrowski, Jarosław Pempera, and Czesław Smutnicki</i> | |
| Genetic-Greedy Hybrid Approach for Topological Active Nets Optimization | 202 |
| <i>José Santos, Óscar Ibáñez, Noelia Barreira, and Manuel G. Penedo</i> | |
| On Sum Coloring of Graphs with Parallel Genetic Algorithms | 211 |
| <i>Zbigniew Kokosiński and Krzysztof Kwarcianny</i> | |
| Liquid State Genetic Programming | 220 |
| <i>Mihai Oltean</i> | |

| | |
|--|-----|
| Genetic Based Distribution Service Restoration with Minimum Average Energy Not Supplied | 230 |
| <i>Thitipong Charuwat and Thanatchai Kulworawanichpong</i> | |
| Multi-objective Feature Selection with NSGA II | 240 |
| <i>Tarek M. Hamdani, Jin-Myung Won, Adel M. Alimi, and Fakhri Karray</i> | |
| Design of 2-D IIR Filters Using Two Error Criteria with Genetic Algorithm | 248 |
| <i>Felicja Wysocka-Schillak</i> | |
| A Hybrid Genetic Algorithm with Simulated Annealing for Nonlinear Blind Equalization Using RBF Networks | 257 |
| <i>Soowhan Han, Imgeun Lee, and Changwook Han</i> | |
| Feature Extraction of Speech Signal by Genetic Algorithms-Simulated Annealing and Comparison with Linear Predictive Coding Based Methods | 266 |
| <i>Melih İnal</i> | |
| Automatic Design of ANNs by Means of GP for Data Mining Tasks: Iris Flower Classification Problem | 276 |
| <i>Daniel Rivero, Juan Rabuñal, Julián Dorado, and Alejandro Pazos</i> | |
| FPGA Implementation of Evolvable Characters Recognizer with Self-adaptive Mutation Rates | 286 |
| <i>Jin Wang, Chang Hao Piao, and Chong Ho Lee</i> | |
| A Multi-gene-Feature-Based Genetic Algorithm for Prediction of Operon | 296 |
| <i>Shuqin Wang, Yan Wang, Wei Du, Fangxun Sun, Xiumei Wang, Yanchun Liang, and Chunguang Zhou</i> | |
| Application of Micro-GA for an Optimal Direct Design Method of Steel Frame | 306 |
| <i>Se-Hyu Choi</i> | |
| Multi-objective Optimal Public Investment: An Extended Model and Genetic Algorithm-Based Case Study | 314 |
| <i>Lei Tian, Liyan Han, and Hai Huang</i> | |
| Particle Swarm Optimization | |
| Many-Objective Particle Swarm Optimization by Gradual Leader Selection | 323 |
| <i>Mario Köppen and Kaori Yoshida</i> | |
| Mixed Ant Colony Optimization for the Unit Commitment Problem | 332 |
| <i>Ana-Talida Serban and Guillaume Sandou</i> | |

| | |
|--|-----|
| A Shuffled Complex Evolution of Particle Swarm Optimization Algorithm | 341 |
| <i>Jiang Yan, Hu Tiesong, Huang Chongchao, Wu Xianing, and Gui Faling</i> | |
| Wasp Swarm Algorithm for Dynamic MAX-SAT Problems | 350 |
| <i>Pedro C. Pinto, Thomas A. Runkler, and João M.C. Sousa</i> | |
| Particle Swarm Optimization for the Multidimensional Knapsack Problem | 358 |
| <i>Fernanda Hembecker, Heitor S. Lopes, and Walter Godoy Jr.</i> | |
| Particle Swarms for Multimodal Optimization | 366 |
| <i>Ender Özcan and Murat Yılmaz</i> | |
| Quantum-Behaved Particle Swarm Optimization with Binary Encoding | 376 |
| <i>Jun Sun, Wenbo Xu, Wei Fang, and Zhilei Chai</i> | |
| Artificial Environment for Simulation of Emergent Behaviour | 386 |
| <i>Rafal Sienkiewicz and Wojciech Jedruch</i> | |
| A Novel and More Efficient Search Strategy of Quantum-Behaved Particle Swarm Optimization | 394 |
| <i>Jun Sun, Choi-Hong Lai, Wenbo Xu, and Zhilei Chai</i> | |
| Learning, Optimization and Games | |
| Extracting Grammars from RNA Sequences | 404 |
| <i>Gabriela Andrejková, Helena Lengeňová, and Michal Matí</i> | |
| Modeling Human Performance in Two Player Zero Sum Games Using Kelly Criterion | 414 |
| <i>Rafal Lopatka and Andrzej Dzieliński</i> | |
| No-Regret Boosting | 422 |
| <i>Anna Gambin and Ewa Szczurek</i> | |
| Evolutionary Approach to the Game of Checkers | 432 |
| <i>Magdalena Kusiak, Karol Wałędzik, and Jacek Mańdziuk</i> | |
| Implementation of an Interactive NPC Based on Game Ontology and Game Community Q/A Bulletin Board | 441 |
| <i>Doo-kyung Park, Tae-bok Yoon, Kyo-hyun Park, Jee-hyong Lee, and Keon-myung Lee</i> | |
| Theory of Saplings Growing Up Algorithm | 450 |
| <i>Ali Karci</i> | |
| Improved Production of Competitive Learning Rules with an Additional Term for Vector Quantization | 461 |
| <i>Enrique Mérida-Casermeyro, Domingo López-Rodríguez, Gloria Galán-Marín, and Juan M. Ortiz-de-Lazcano-Lobato</i> | |

| | |
|---|-----|
| Reinforcement Learning in Fine Time Discretization | 470 |
| <i>Paweł Wawrzyński</i> | |
| Agent-Based Approach to Solving the Resource Constrained Project Scheduling Problem | 480 |
| <i>Piotr Jedrzejowicz and Ewa Ratajczak-Ropel</i> | |
| A Model of Non-elemental Associative Learning in the Mushroom Body Neuropil of the Insect Brain | 488 |
| <i>Jan Wessnitzer, Barbara Webb, and Darren Smith</i> | |
| Performance-Based Bayesian Learning for Resource Collaboration Optimization in Manufacturing Grid | 498 |
| <i>Jian Zhou, Qing Li, Jim Browne, Qing Wang, Paul Folan, and TianYuan Xiao</i> | |
| A Hybrid Simulated-Annealing Algorithm for Two-Dimensional Strip Packing Problem | 508 |
| <i>Türkay Dereh and Gülesin Sena Daş</i> | |
| Handling Linguistic Values in Knowledge Acquisition | 517 |
| <i>Dae-Young Choi</i> | |
| An IA Based Approach for the Optimal Design of Traffic-Monitor Systems | 526 |
| <i>Yi-Chih Hsieh, Yung-Cheng Lee, and Ta-Cheng Chen</i> | |
| Finding the Optimal Path in 3D Spaces Using EDAs – The Wireless Sensor Networks Scenario | 536 |
| <i>Bo Yuan, Maria Orlowska, and Shazia Sadiq</i> | |
| Evidential Reasoning Based on Multisensor Data Fusion for Target Identification | 546 |
| <i>Xin Wang, Yunxiao Wang, Xiao Yu, Zhengxuan Wang, and Yunjie Pang</i> | |
| A Simple and Compact Algorithm for the RMQ and Its Application to the Longest Common Repeat Problem | 554 |
| <i>Inbok Lee and Ha Yoon Song</i> | |
| Improved Bacterial Foraging Algorithms and Their Applications to Job Shop Scheduling Problems | 562 |
| <i>Chunguo Wu, Na Zhang, Jingqing Jiang, Jinhui Yang, and Yanchun Liang</i> | |

Fuzzy and Rough Systems

| | |
|---|-----|
| An Evolutionary Approach for Approximating the Solutions of Systems of Linear Fuzzy Equations | 570 |
| <i>Nguyen Hoang Viet and Michał Kleiber</i> | |
| On Fuzzy Driven Support for SD-Efficient Portfolio Selection | 578 |
| <i>Włodzimierz Ogryczak and Andrzej Romaszkiwicz</i> | |

| | |
|--|-----|
| Fuzzy Kernel Ridge Regression for Classification | 588 |
| <i>YoungSik Choi and JiSung Noh</i> | |
| Assessment of the Accuracy of the Process of Ceramics Grinding with the Use of Fuzzy Interference | 596 |
| <i>Dariusz Lipiński and Wojciech Kacalak</i> | |
| A Dynamic Resource Broker and Fuzzy Logic Based Scheduling Algorithm in Grid Environment | 604 |
| <i>Jiayi Zhou, Kun-Ming Yu, Chih-Hsun Chou, Li-An Yang, and Zhi-Jie Luo</i> | |
| Improving Business Failure Predication Using Rough Sets with Non-financial Variables | 614 |
| <i>Jao-Hong Cheng, Chung-Hsing Yeh, and Yuh-Wen Chiu</i> | |
| Optimization of Fuzzy Model Driven to IG and HFC-Based GAs | 622 |
| <i>Jeoung-Nae Choi, Sung-Kwun Oh, and Hyung-Soo Hwang</i> | |
| Potential Assessment of an Ellipsoidal Neural Fuzzy Time Series Model for Freeway Traffic Prediction | 631 |
| <i>Ping-Feng Pai, Kuo-Ping Lin, and Ping-Teng Chang</i> | |
| Digital Model of Series Resonant Converter with Piezoelectric Ceramic Transducers and Fuzzy Logic Control | 640 |
| <i>Pawel Fabijański and Ryszard Lagoda</i> | |
| A Method to Classify Collaboration in CSCL Systems | 649 |
| <i>Rafael Duque and Crescencio Bravo</i> | |
| Electromagnetic Levitation System with Clustering Based Fuzzy Controller | 657 |
| <i>Min-Soo Kim and Yeun-Sub Byun</i> | |
| Fuzzy Relation-Based PNNs with the Aid of IG and Symbolic Gene Type-Based GAs | 666 |
| <i>Sung-Kwun Oh, In-Tae Lee, Hyun-Ki Kim, and Seong-Whan Jang</i> | |
| Pricing the Foreign Currency Options with the Fuzzy Numbers Based on the Garman-Kohlhagen Model | 674 |
| <i>Fan-Yong Liu</i> | |
| Designing Rough Sets Attributes Reduction Based Video Deinterlacing System | 684 |
| <i>Gwanggil Jeon, Marco Anisetti, Valerio Bellandi, and Jechang Jeong</i> | |
| Optimization of Fuzzy Membership Function Using Clonal Selection | 694 |
| <i>Ayşe Merve Şakiroğlu and Ahmet Arslan</i> | |
| Classification and Clustering | |
| Clustering of Leaf-Labelled Trees | 702 |
| <i>Jakub Koperwas and Krzysztof Walczak</i> | |

| | |
|---|-----|
| Social Organization of Evolving Multiple Classifier System Functioning in Changing Environments | 711 |
| <i>Sarunas Raudys</i> | |
| Softening Splits in Decision Trees Using Simulated Annealing | 721 |
| <i>Jakub Dvořák and Petr Savický</i> | |
| A Novel Architecture for the Classification and Visualization of Sequential Data | 730 |
| <i>Jorge Couchet, Enrique Ferreira, André Fonseca, and Daniel Manrique</i> | |
| Locally Scaled Density Based Clustering | 739 |
| <i>Ergun Biçici and Deniz Yuret</i> | |
| Hierarchical Rules for a Hierarchical Classifier | 749 |
| <i>Igor T. Podolak</i> | |
| A Demonstration of Clustering in Protein Contact Maps for Alpha Helix Pairs | 758 |
| <i>Robert Fraser and Janice Glasgow</i> | |
| Dynamic Data Probes | 767 |
| <i>David W. Pearson</i> | |
| Classifying Chemical Compounds Using Contrast and Common Patterns | 772 |
| <i>Andrzej Dominik, Zbigniew Walczak, and Jacek Wojciechowski</i> | |
| Power Transients Characterization and Classification Using Higher-Order Cumulants and Competitive Layers | 782 |
| <i>Juan-José González de-la-Rosa, Antonio Moreno Muñoz, Isidro Lloret, Carlos G. Puntonet, and Juan-Manuel Górriz</i> | |
| Mutual Information Estimation in Higher Dimensions: A Speed-Up of a k -Nearest Neighbor Based Estimator | 790 |
| <i>Martin Vejmelka and Kateřina Hlaváčková-Schindler</i> | |
| Grammar-Based Classifier System for Recognition of Promoter Regions | 798 |
| <i>Olgierd Unold</i> | |
| Learning Bayesian Classifiers from Dependency Network Classifiers | 806 |
| <i>José A. Gámez, Juan L. Mateo, and José M. Puerta</i> | |
| Determining the Dependency Among Clauses Based on Machine Learning Techniques | 814 |
| <i>Mi-Young Kim</i> | |
| Using Real-Valued Meta Classifiers to Integrate and Contextualize Binding Site Predictions | 822 |
| <i>Mark Robinson, Offer Sharabi, Yi Sun, Rod Adams, Rene te Boekhorst, Alistair G. Rust, and Neil Davey</i> | |

| | |
|---|-----|
| Effectiveness of Feature Space Selection on Credit Engineering on Multi-group Classification Cases | 830 |
| <i>Junghee Park, Kidong Lee, and Jinhwa Kim</i> | |
| Constructing Stereotypes for an Adaptive e-Shop Using AIN-Based Clustering | 837 |
| <i>Maria Virvou, Anastasios Savvopoulos, George A. Tsihrintzis, and Dionisos N. Sotiropoulos</i> | |
| Author Index | 847 |

Evolutionary Induction of Decision Trees for Misclassification Cost Minimization

Marek Krętowski and Marek Grześ

Faculty of Computer Science, Białystok Technical University
Wiejska 45a, 15-351 Białystok, Poland
{mkret,marekg}@ii.pb.bialystok.pl

Abstract. In the paper, a new method of decision tree learning for cost-sensitive classification is presented. In contrast to the traditional greedy top-down inducer in the proposed approach optimal trees are searched in a global manner by using an evolutionary algorithm (EA). Specialized genetic operators are applied to modify both the tree structure and tests in non-terminal nodes. A suitably defined fitness function enables the algorithm to minimize the misclassification cost instead of the number of classification errors. The performance of the EA-based method is compared to three well-recognized algorithms on real-life problems with known and randomly generated cost-matrices. Obtained results show that the proposed approach is competitive both in terms of misclassification cost and compactness of the classifier at least for some datasets.

1 Introduction

Nowadays computer-aided decision support systems become more and more popular in solving complex decision-making problems in marketing, finance and medicine. Based on gathered datasets of examples they enable training various types of classifiers in form of neural networks, decision trees and rules. However, in most cases only the number of classification errors is taken into account during the induction process. In many practical applications this classical approach is not suitable because there are other factors, such as costs, which may influence final decisions. In [24] Turney discussed different types of costs associated with inductive learning (e.g., the cost of tests, the cost of objects and the misclassification cost). The term *cost-sensitive classification* encompasses all these types of costs.

There are two main groups of methods for making a classifier cost-sensitive. In the first group, individual error-based systems are converted into cost-sensitive ones. One of the first attempts to incorporate misclassification costs into decision tree learning was made in the *CART* system [4]. The method consists in the modification of the class prior probabilities used in the splitting criterion and in the application of the cost-based measure to a tree pruning. Another cost-sensitive methods for pruning decision trees are proposed in [10,3]. However, it should be emphasized that stand-alone pruning procedures have only a limited capability to change the tree structure created by an error-based inducer. In [22]

the *C4.5* system [21] was modified using instance-weighting, but the method requires converting of a cost matrix into a cost vector, which can result in poor performance in multi-class problems. Recently, Ling *et al.* [17,26] proposed an algorithm that minimizes the sum of the misclassification and test costs. This approach is based on a new splitting criterion (*total cost*) for nominal attributes and two-class problems.

The second group includes general methods for making an arbitrary classifier cost-sensitive. *MetaCost* [7] is based on wrapping a meta-learning stage around the error-based classifier. Another method proposed by Zadrozny *et al.* [25] uses cost-proportionate rejection sampling and ensemble aggregation. Iterative weighting and gradient boosting are investigated by Abe *et al.* [1] in multi-class problems.

The proposed approach consists in developing a specialized evolutionary algorithm for global induction of cost-sensitive decision tree classifiers. Several EA-based systems which learn decision trees in the top-down manner (e.g. *BTGA* [6], *OC1-ES* [5], *DDT-EA* [12]) have been proposed so far. Generally, they apply evolutionary approach to the search of splitting hyper-planes in non-terminal nodes of oblique decision trees.

In this paper, a global approach to decision tree induction is advocated. In contrast to a step-wise construction, the whole tree is being searched at the time. It means the simultaneous search for an optimal structure of the tree and for all tests in non-terminal nodes. The global approach was initially proposed by Koza in [11], where genetic programming was used for evolving LISP S-expressions that correspond to simple decision trees. A similar idea was investigated in the *GATree* system [20] which directly evolves classification trees with nominal tests. In our previous papers, we showed that EA-based global inducer can efficiently generate univariate [13], linear [14] and mixed [15] decision trees.

Concerning applications of evolutionary techniques to cost-sensitive learning of decision trees, according to our knowledge, only one attempt can be mentioned. In [23] Turney proposed the *ICET* system, which uses the standard genetic algorithm to evolve a population of biases for the modified *C4.5*. Both feature and misclassification costs are taken into account.

The rest of the paper is organized as follows. In the next section, the proposed evolutionary algorithm is described in details. Section 3 contains experimental validation of the approach on the benchmark classification problems with known and randomly generated cost-matrices. In the last section conclusions and possible directions of the future work are presented.

2 Evolutionary Algorithm for Global Induction

The structure of the proposed evolutionary algorithm follows the typical framework [18] and only application-specific issues (the fitness function, specialized genetic operators, ...) are described in more detail in this section.

2.1 Preliminaries

We assume that a learning set $E = \{e_1, e_2, \dots, e_M\}$ consists of M examples. Each example $e \in E$ is described by N attributes (features) A_1, A_2, \dots, A_N and labeled by a class $c(e) \in C$. The set of all examples from the class $c_k \in C$ is denoted by $C_k = \{e \in E : c(e) = c_k\}$ and the class assigned (predicted) by the tree T to the example e is denoted by $T(e)$.

Let $Cost(c_i, c_j) \geq 0$ be the cost of misclassifying an object from the class c_j as belonging to the class c_i . We assume that the cost of the correct decision is equal zero i.e., $Cost(c_i, c_i) = 0$ for all c_i .

2.2 Representation, Initialization and Termination Condition

In our system, decision trees are represented in their actual form as classical univariate trees where each test in a non-terminal node concerns only one attribute. Additionally, in every node information about learning vectors associated with the node is stored and it enables the algorithm to perform efficiently local structure and tests modifications during applications of genetic operators.

In case of a nominal attribute at least one value is associated with each branch. It means that an inner disjunction is built-in into the induction algorithm. For a continuous-valued feature typical inequality tests with boundary thresholds¹ as potential splits are considered. All boundary thresholds for each continuous-valued attribute are calculated before starting the evolutionary induction [13]. It significantly limits the number of possible splits and focuses the search process.

In standard error-based decision trees class labels are associated with leaves by using the majority rule based on training objects which reached a leaf-node. In cost-sensitive case class labels for leaves are chosen to minimize the misclassification cost in each leaf.

Individuals in the initial population are generated as follows. The classical top-down algorithm is applied, but tests are chosen in a dipolar way [12]. Among feature vectors located in the considered node two objects from different classes (so called *mixed dipole*) are randomly chosen. An effective test, which separates two objects into sub-trees, is created randomly by taking into account only attributes with different feature values. Recursive divisions are repeated until the stopping condition (based on the minimal number of learning vectors in a node or homogeneity of a node) is met. Finally, the resulting tree is post-pruned according to the fitness function.

The algorithm terminates if the fitness of the best individual in the population does not improve during the fixed number of generations (default value is equal 200). Additionally, the maximum number of generations is specified which limits the computation time in case of a very slow convergence (default value: 1000).

¹ A boundary threshold for the given attribute is defined as a midpoint between such a successive pair of examples in the sequence sorted by the increasing value of the attribute, in which the examples belong to two different classes.

2.3 Fitness Function

The properly defined fitness function is a crucial element for every evolutionary algorithm. In case of induction of decision structures it is obvious that there is no possibility to directly optimize accuracy of the classifier on unseen examples. Instead, the system performance on the training data is usually used to guide the search process and additional factors are introduced to prevent the overfitting and to increase the generalization power of the classifier (e.g. [13,16]). An analogous approach is applied in our system.

The misclassification cost $MC(T)$ of the tree T is estimated on the training data:

$$MC(T) = \frac{1}{M} \cdot \sum_{e \in E} Cost(T(e), c(e)). \quad (1)$$

The values of the misclassification cost do not fit into range $[0, 1]$ like classification errors, however, it is easy to calculate the maximal cost for a given dataset and a cost matrix:

$$MaxMC = \frac{1}{M} \cdot \sum_{c_k \in C} |C_k| \cdot \max_{i \neq k} Cost(c_i, c_k). \quad (2)$$

Hence, by dividing $MC(T)$ by $MaxMC$ one can obtain the *normalized misclassification cost*, which is equal 0 for the perfect prediction and 1 in the worst case. Finally, the fitness function, which is maximized, is defined as follows:

$$Fitness(T) = \left(1 - \frac{MC(T)}{MaxMC}\right) \cdot \frac{1}{1 + \alpha \cdot S(T)}, \quad (3)$$

where $S(T)$ is the size of the tree T expressed as a number of nodes and α is a user supplied parameter (default value is 0.001).

It should be expected that there is no one optimal value of α for all possible datasets and cost matrices. When the certain problem is analyzed, tuning this parameter may lead to the improvement of the results (in terms of the misclassification cost or classifier complexity).

2.4 Genetic Operators

There are two specialized genetic operators called *CrossTrees* and *MutateNode* which fulfill the role of crossover and mutation operators from the classic framework.

CrossTrees alike the standard crossover operator modifies two chromosomes (i.e. trees) by swapping their certain parts. There are three types of crossover-like modifications that can be performed on trees: two of them concern sub-trees and one only tests. Firstly, one random node is drawn from each tree. Then the type of modification is chosen. By default all types of these operations are equally probable, but the user can specify his own proportions. First one exchanges sub-trees rooted in chosen nodes. This variant is analogous to the typical crossover operator introduced in genetic programming. Second operator replaces only tests

between chosen nodes. This type of modification is possible solely when tests have the same number of outcomes. Third operator is the most radical. It replaces in random order all branches of chosen nodes. It is evident that the same effect can be achieved by combining two or more exchanges of the first type.

MutateNode, like the standard mutation operator, takes one argument (i.e. a single node of the tree). This operator can be applied to each node with a given probability (default value 0.05). The result of this operator depends on what kind of a node is considered (i.e. a leaf or an internal node). When modifying an internal node the following possibilities are by default equally probable (this probabilities are again user specified):

- a test in the current node can be replaced by a completely new test chosen in a dipolar way,
- a less drastic possibility in reference to the previous one consists in changing the threshold of a test on a real attribute or modifying groups of nominal values (i.e. two branches of the current node can be merged into one branch or a group can be split creating an additional branch) according to the application of an inner disjunction to tests on nominal attributes,
- a test in the current node and a test from one of node's sons can be exchanged, it concerns nodes which have non-leaf descendants
- one of the branches of the current node can be multiplied and replace another branch which is to be dropped,
- each node, even the root, can be transformed into a leaf; this operator allows reducing in a straight way the tree size.

A leaf node can be modified on condition that it contains feature vectors belonging to different classes. Such a leaf can be replaced by:

- a non-terminal node with a randomly chosen test,
- a sub-tree generated according to the dipolar algorithm which is also applied during initialization.

After application of some of the described operators it may be necessary to alter locations of some of the learning vectors. It can lead to such a situation where there are nodes or even whole sub-trees without any training examples. For this reason empty parts of the tree have to be removed. It is done by using either simple algorithm which does it directly or by specialized operator called *MaximizeFitness*. This operator not only drops empty parts of the tree but also performs more sophisticated modifications. It visits all nodes of a tree in bottom-up order. It tries to replace each not-terminal node by a leaf while taking into account potential gain in the fitness. *MaximizeFitness* is by default applied to all trees from initial population and to sub-trees which appear after replacing leaves.

As a selection mechanism the ranking linear selection [18] is applied. Additionally, the chromosome with the highest value of the fitness function in the iteration is copied to the next population (the *elitist strategy*).

3 Experimental Results

In this section experimental validation is presented. The proposed approach (denoted in tables as *GDT-MC*) is compared with the commercial cost-sensitive classifier *C5.0* which is enhanced version of *C4.5* [21]. Our global inducer is also compared with *MetaCost* [7] and *CostSensitiveClassifier* (denoted as *CSJ48*) of the *Weka* system [9]. Both *MetaCost* and *CSJ48* are based on wrapping an error-based classifier (*J48* which is *Weka*'s implementation of *C4.5* was used in the experiments).

Performance of all presented systems is assessed on a subset of the well-known datasets publicly available from the *UCI Machine Learning Repository* [2]. More complex datasets with continuous-valued features and no missing values were chosen. All results presented in the tables correspond to averages of 10 runs and were obtained by using test sets (when available) or by the 10-fold stratified crossvalidation. The average number of leaves is given as a complexity measure of classifiers.

3.1 Known Cost-Matrices

Only for two datasets (namely *german* and *heart*) misclassification costs are provided. The cost matrix in both cases is the same and non-zero costs are as follows: $Cost(c_2, c_1) = 1$ and $Cost(c_1, c_2) = 5$.

Table 1. Misclassification costs and tree sizes obtained for datasets with known cost matrix

| | <i>MetaCost</i> | | <i>CSJ48</i> | | <i>C5.0</i> | | <i>GDT-MC</i> | |
|---------------|-----------------|-------|--------------|-------|-------------|-------|---------------|-------|
| Dataset | Cost | Size | Cost | Size | Cost | Size | Cost | Size |
| <i>german</i> | 1.26 | 31.61 | 0.69 | 67.31 | 0.71 | 81.37 | 0.58 | 11.69 |
| <i>heart</i> | 1.01 | 18.05 | 0.52 | 10.87 | 0.56 | 16.49 | 0.61 | 25.07 |
| average | 1.14 | 24.83 | 0.61 | 39.09 | 0.64 | 48.93 | 0.6 | 18.38 |

The results obtained with the default value of α are collected in Table 1. It should be noted that in case of the *german* dataset EA-based system performs better (both in terms of the misclassification cost and the classifier size) than all remaining algorithms. As for the second dataset *GDT-MC* is slightly worse than *C5.0* and *CSJ48* but still much better than *MetaCost*.

In order to verify the impact of the α parameter on the results, a series of experiments with varying α was prepared (see Fig. 1 and Fig. 2). As it could be expected, along with a decrease of α an increase of trees complexity can be observed. Concerning the misclassification cost, after initial small decrease global minimum is reached and the cost quickly rises for larger trees. It can be also observed that for both datasets the default setting of this parameter (denoted by vertical dotted line) is not really optimal. For the *german* dataset, where *GDT-MC* obtained the best result (0.58) among all competitors,

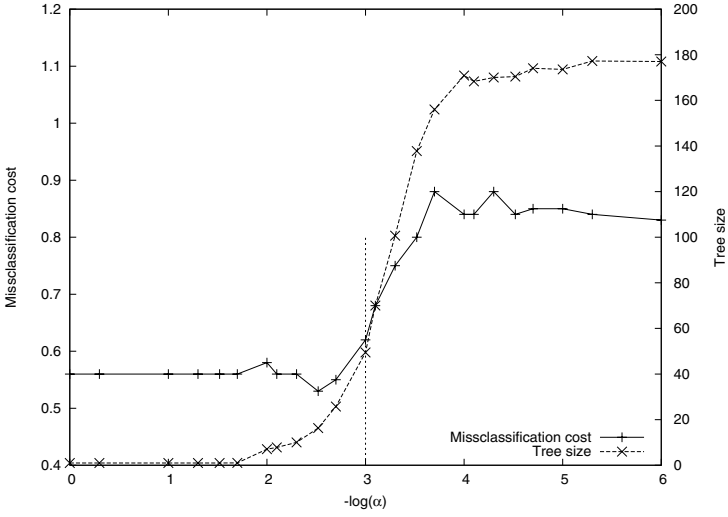


Fig. 1. The impact of α parameter on the misclassification cost and the tree complexity for *heart* dataset

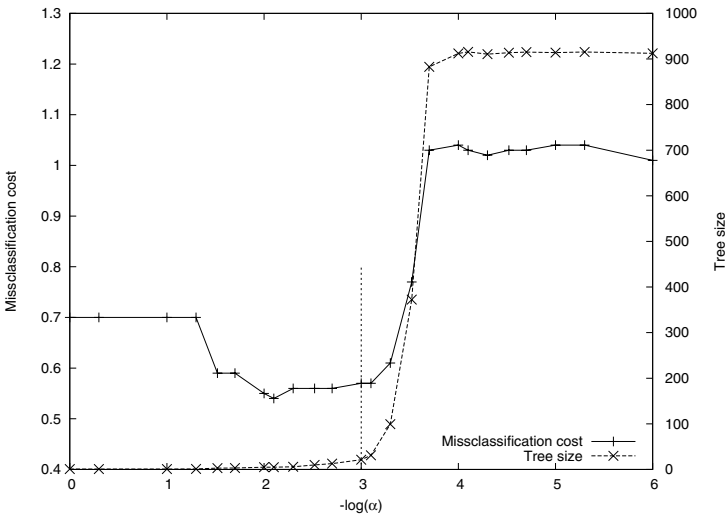


Fig. 2. The impact of α parameter on the misclassification cost and the tree complexity for *german* dataset

a further decrease of the misclassification cost to 0.54 is possible. Concerning the *heart* dataset it is also possible to achieve the misclassification cost lower than these obtained by *C5.0* or *CSJ48*.

3.2 Simulated Cost-Matrices

For the remaining datasets, for which cost matrices are not provided, a different but typical experimental setup is applied (see e.g. [22,19,16]). In each run of the 10-fold crossvalidation a cost matrix was generated randomly. The off-diagonal elements of the cost matrix were drawn from the uniform distribution over the range [1, 10]. The diagonal elements were always zero. The standard deviations are not presented because if they had been calculated they would have expressed the effects of varying cost matrices [7]. For each single crossvalidation run the same random cost matrix and the same training data splits were used for all tested algorithms.

Table 2. The average misclassification cost and the tree size for datasets with randomly generated cost matrices

| | <i>MetaCost</i> | | <i>CSJ48</i> | | <i>C5.0</i> | | <i>GDT-MC</i> | |
|------------------------|-----------------|-------|--------------|-------|-------------|-------|---------------|-------|
| Dataset | Cost | Size | Cost | Size | Cost | Size | Cost | Size |
| <i>breast - w</i> | 0.29 | 10.02 | 0.24 | 9.81 | 0.24 | 11.15 | 0.22 | 5.6 |
| <i>australian</i> | 0.63 | 7.0 | 0.40 | 8.0 | 0.60 | 12.50 | 0.64 | 12.0 |
| <i>balance - scale</i> | 1.33 | 34.0 | 1.40 | 19.0 | 1.27 | 29.80 | 1.16 | 22.4 |
| <i>bupa</i> | 2.44 | 22.57 | 1.61 | 14.94 | 1.59 | 17.02 | 1.70 | 43.76 |
| <i>cars</i> | 0.21 | 31.0 | 0.07 | 31.0 | 0.05 | 25.60 | 0.05 | 30.0 |
| <i>cmc</i> | 3.08 | 148.6 | 2.46 | 129.0 | 2.36 | 166.3 | 2.04 | 9.87 |
| <i>glass</i> | 2.40 | 14.0 | 2.40 | 14.0 | 1.64 | 20.70 | 1.69 | 33.1 |
| <i>page - blocks</i> | 0.19 | 41.09 | 0.17 | 36.57 | 0.17 | 34.57 | 0.24 | 4.33 |
| <i>pima</i> | 1.64 | 28.19 | 0.93 | 11.82 | 0.93 | 15.61 | 0.91 | 8.92 |
| <i>wine</i> | 1.08 | 5.0 | 1.08 | 5.0 | 0.88 | 5.10 | 0.78 | 6.40 |
| <i>vehicle</i> | 1.96 | 70.88 | 1.53 | 57.21 | 1.53 | 68.98 | 1.54 | 16.69 |
| average | 1.39 | 37.49 | 1.12 | 30.58 | 1.02 | 37.03 | 1.0 | 17.55 |

As it could be observed in Table 2, for 5 out of 11 datasets, misclassification costs of the classifiers generated by the proposed method are lower than costs of the competitors. It is an indicative achievement while taking into account the fact that it was compared with so many renowned and efficient counterparts. On the other hand it is worth mentioning that there is only one dataset - *page blocks* - on which *GDT-MC* is slightly worse than all remaining algorithms. But it was verified that tuning the parameter α leads to the real improvement: 0.11 ($\alpha = 0.00005$) which gives *GDT-MC* the best score.

Finally, it was confirmed one more time that global induction generally results in less complex decision structures than obtained by top-down inducers. Only for *bupa* dataset the resulting decision tree was overgrown.

It should be mentioned that the global induction requires more processing time compared to traditional top-down algorithms. Nevertheless, the learning time required by *GDT-MC* is acceptable. For instance the system needs 2 minutes and 20 second (on average) of CPU time on a PC workstation (PIV 3GHz, 1GB RAM) to generate classifier for the largest dataset (*page blocks* - 5473 examples, 10 features and 5 classes).

4 Conclusions

In the paper, a new method of univariate decision tree induction for misclassification cost minimization is presented. The proposed approach consists in developing specialized evolutionary algorithm which globally searches for optimal decision tree. Results of the experimental validation show that our system is able to generate competitive classifiers both in terms of the misclassification cost and the decision tree size. It should be also noted that by tuning the α parameter which corresponds to the importance of the complexity term in the fitness function, even better results for the given dataset can be obtained.

Furthermore, many possibilities for improvement still exist (e.g. better fitness function, new genetic operators, ...). One direction of current research is an extension of the cost model by incorporating costs of features (tests). It can be done mainly by modifying the fitness function.

The proposed approach is not the fastest one now, but hopefully it is well-known that evolutionary algorithms are well-suited for parallel architecture. We plan to speed up our system by re-implementing it in the distributed environment.

Acknowledgments. This work was supported by the grant W/WI/5/05 from Białystok Technical University.

References

1. Abe, N., Zadrozny, B., Langford, J.: An iterative method for multi-class cost-sensitive learning. In *KDD'04*. ACM Press, (2004) 3–11.
2. Blake, C., Keogh, E., Merz, C.: *UCI repository of machine learning databases*, [<http://www.ics.uci.edu/~mlearn/MLRepository.html>], (1998).
3. Bradford, J.P., Kunz, C., Kohavi, R., Brunk, C., Brodley, C.E.: Pruning decision trees with misclassification costs. In *Proc. of ECML'98*. Springer (1998) 131–136.
4. Breiman, L., Friedman, J., Olshen, R., Stone C.: *Classification and Regression Trees*. Wadsworth Int. Group (1984).
5. Cantu-Paz, E., Kamath, C.: Inducing oblique decision trees with evolutionary algorithms. *IEEE Transactions on Evolutionary Computation* **7**(1) (2003) 54–68.
6. Chai, B. *et al.*: Piecewise-linear classifiers using binary tree structure and genetic algorithm. *Pattern Recognition* **29**(11) (1996) 1905–1917.
7. Domingos, P.: MetaCost: A general method for making classifiers cost-sensitive. In *Proc. of KDD'99.*, ACM Press (1999) 155–164.
8. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In *Proc. of IJCAI'93.*, (1993) 1022–1027.

9. Frank, E. *et al.*: Weka 3 - Data Mining with Open Source Machine Learning Software in Java. [<http://www.cs.waikato.ac.nz/~ml/weka>]. University of Waikato (2000).
10. Knoll, U., Nakhaeizadeh, G., Tausend, B.: Cost-sensitive pruning of decision trees. *LNCS* 784 (1994) 383–386.
11. Koza, J.: Concept formation and decision tree induction using genetic programming paradigm. *LNCS* 496 (1991) 124–128.
12. Krętownski, M.: An evolutionary algorithm for oblique decision tree induction. *LNAI* 3070, (2004) 432–437.
13. Krętownski, M., Grześ, M.: Global learning of decision trees by an evolutionary algorithm. In: *Information Processing and Security Systems*, Springer, (2005) 401–410.
14. Krętownski, M., Grześ, M.: Evolutionary learning of linear trees with embedded feature selection. *LNAI* 4029, (2006) 400–409.
15. Krętownski, M., Grześ, M.: Mixed decision trees: An evolutionary approach. *LNCS* 4081, (2006) 260–269.
16. Kwedlo, W., Krętownski, M.: An evolutionary algorithm for cost-sensitive decision rule learning. *LNAI* 2167, (2001) 288–299.
17. Ling, C., Yang, Q., Wang, J., Zhang, S.: Decision trees with minimal costs, In: *Proc. of ICML'04.*, ACM Press (2004), Article No. 69.
18. Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. 3rd edn. Springer (1996).
19. Margineantu, D.D., Dietterich, T.G.: Bootstrap methods for the cost-sensitive evaluation of classifiers. In *Proc. of ICML'2000.*, Morgan Kaufmann (2000) 583–590.
20. Papagelis, A., Kalles, D.: Breeding decision trees using evolutionary techniques. In: *Proc. of ICML'01.*, Morgan Kaufmann (2001) 393–400.
21. Quinlan, J.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann (1993).
22. Ting, K.M.: An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering* **14**(3), (2002) 659–665.
23. Turney, P.: Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research* **2** (1995) 369–409.
24. Turney, P.: Types of cost in inductive concept learning. In *Proc. of ICML'2000 Workshop on Cost-Sensitive Learning*. Stanford, CA (2000).
25. Zadrozny, B., Langford, J., Abe, N.: Cost-sensitive learning by cost-proportionate example weighting concept learning. In *Proc. of ICDM'03*. IEEE Press (2003).
26. Zhang S., Qin, Z., Ling, C. Sheng, S.: Missing is usefull: Missing values in cost-sensitive decision trees. *IEEE Transactions on Knowledge and Data Engineering* **17**(12), (2005) 1689–1693.

DNA Based Evolutionary Approach for Microprocessor Design Automation

Nagarajan Venkateswaran¹, Arjun Kumeresh², and Harish Chandran²

¹ Director, WARan Research FoundaTion (WARFT), Chennai, India

² Research Trainees WARFT

{waran, arjun, harish}@warftindia.org

Abstract. In a paper [1] presented to BICS 2006, a basic methodology for microprocessor design automation using DNA sequences was proposed. A refined methodology with new schemes for traversal, encoding, recombination, and processor evaluation are proposed in this paper. Moreover concepts such as mutation, graphical decoding and environment simulation are introduced and a new technique for creating DNA based algorithms used in the mutation process is also presented. The proposed methodology is then generalized to extend its application to other domains. This paper presents a conceptual framework whose implementation aspects are still under investigation.

1 Introduction

Conventional microprocessor design automation involves optimization at various phases to minimize gate count, power, chip area and to maximize performance. Many tools are available to automate these processes. Evolutionary algorithms are applied towards these optimization processes. This is due to the fact that evolution is an optimization process by which the phenotype of a population gets optimized over successive generations. However, the functional level architectural design is not a part of the automation. Rather the architecture is created by a team of architects using their prior experience of designing various microprocessors. The fact that evolution has produced complex organisms such as humans stands testimony to its success as an optimization process. Natural evolution involves DNA based processes. But modeling this natural evolution at the DNA level with realistic encoding and recombination processes has not been attempted to automate microprocessor design [2]. In fact, microprocessors, which are less complex than humans, can be evolved naturally without human intervention if their characteristics (phenotype) can be mapped onto the DNA domain through a biologically realistic encoding process. Then automating microprocessor design process would reduce to combining different microprocessor DNAs to produce an offspring and simulating their working environment thereby incorporating random variation and selection which form the two major steps in natural evolution. This paper proposes such a DNA based approach for automating microprocessor design (for both general purpose as well as ASIC) which involves automation at all levels from the functional level to the layout level.

2 The Methodology

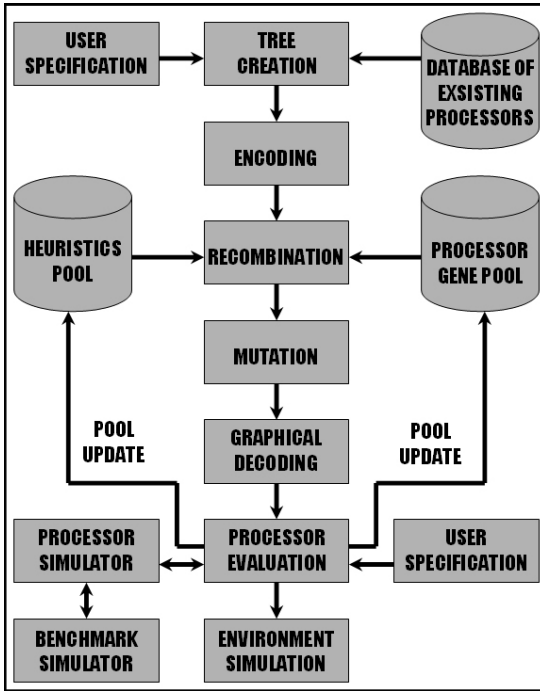


Fig. 1. The Proposed Methodology

Fig. 1 shows the proposed methodology. Specifications of the desired processor (*User Specifications*) are obtained from the user. The user can mention the power and performance ratings required from the processor and also its functionalities. A typical spec could be: 32 bit ALU based General Purpose processor rated at 3 Gigaflops at 80 Watts. An example of an ASIC spec could be: 32 bit processor with high fault tolerance, for image processing applications performing at 2 Gigaflops at 20 Watts. A *Database of Existing Processors* is built up beforehand. If an appropriate match is found, then that design is given to the user. Else, processors having similar characteristics are chosen and their *Parameter Trees* are built.

‘Characteristics’ of a processor refer to those parameters which completely and uniquely describe a processor. These parameters, such as clock speed, netlist density, threshold voltage, on chip memory etc., play a vital role in the design and working of the processor. The DNA sequences of the processors are then obtained using a specific, nature inspired *Encoding* scheme. *Recombinations* between various sequences are driven by certain heuristics. Certain components of a processor like Carry Save Adder, Baugh-Wooley Multiplier etc are pre-encoded and stored in the *Gene Pool*. These components can then be added to the offspring processor based on the user specifications. Moreover, processors evolved in previous iterations are also stored in the gene pool. *Mutation or Post Processing Stage I* involves the extraction of the overall netlist of the processor. *Graphical Decoding* is done to fix the layout level map of the processor and incorporates *Embryogenesis*. Next, the processor is evaluated using a *Processor Simulator* that runs a *Benchmarking Suite* on it. Minor deviations from the user specifications are corrected in *Environment Simulation* or *Post Processing Stage II*. Here, environment refers to a set of applications that are to be run on the processor. Finally, after the processor is tested and evaluated, the goodness of each heuristic used in its creation is updated. If a satisfactory processor is found, then the gene pool is also updated with that processor. The following sections elaborate on these processes.

3 Tree Creation and Traversal

The first step in the methodology involves mapping the characteristics of a processor onto a *Directed Acyclic Graph* (parameter tree) where each node represents a characteristic and the edge weight between two nodes quantifies the dependency between the nodes.

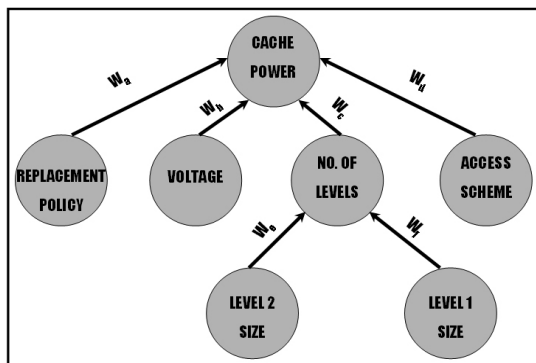


Fig. 2 shows a part of the parameter tree of a cache structure. According to this tree, the power consumed by the cache depends on 4 parameters. Of these, *No. Of Levels* has 2 children. This is a direct consequence of the fact that the values of the parameter *Level 1 Size* and *Level 2 Size* is not 0. The edges in the figure have weights $W_a, W_b \dots W_f$.

Fig. 2. Example: Part of a Cache Parameter Tree

These weights signify the level of dependence between two nodes: higher the weight, higher the dependence. For example, *Cache Power* is most affected by *Voltage*. Hence W_b will be greater than W_a, W_c and W_d . These weights play a vital role while traversing the tree.

To build the parameter tree, various characteristics that define a processor must be obtained. Then various relationships among these characteristics must be studied. This can be done by collecting a large number of processor specifications and observing the trend among various parameters. For example, if most of the processors show that increasing the cache levels increases the power consumption tremendously, a high edge weight can be associated between these two parameters. A simple data mining algorithm can automate this process.

The Node Function of a particular node is a mathematical relation connecting the *Node Data* of the node to the *Node Data* of its parents and children. Fig. 3 shows a simple functional dependence. The parameter MNIPC (Maximum Number of Instruction Per Clock-cycle) is dependent on 3 of its parents: the No. of Cores, the NDP (Number of Data Paths) per core and the Number of Processors. MNIPC is just the direct product of these values. The node function is very important because it defines the actual mathematical relations between nodes. Thus node values can be fixed up and validated during recombination. For example, if the user specifies a MNIPC of 6, then we can fix the No. of Cores, No. of Processors and NDP as any combination of 1, 2, 3 ($1 \times 2 \times 3 = 6$). This could be one of the heuristics during recombination.

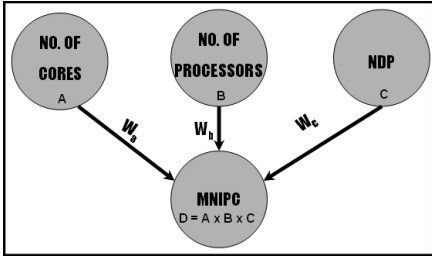


Fig. 3. The Node Function

Fixing these functions can be quite difficult as complete knowledge of processor architecture is required to manually find the functions. But it is possible to automate this process of finding the node function using supervised learning algorithms. For this, large number of processors must be analyzed and their trees must be created.

By applying this process to non connected nodes, inferences can be drawn about the way they interact indirectly. For example, instruction size and type of full adder used in the ALU don't have any obvious relationship. But if hidden relations do exist between them, then they could be found out using this method. The node function can be stored in standard formats like post-fix notations with variables with pointers to other nodes. The node function is mostly architecture independent. But, sometimes, certain architectures may change the function to an extent. For example, the inclusion of *Hyper-threading* may change the equation governing *Level 1 Cache Size*.

A *Depth First Traversal* (DFT) is performed on the parameter tree to build a linear sequence of nodes such that related nodes are placed as close to each other as possible. In natural DNA sequences, related genes are placed close to each other. Since DFT visits a node that is most strongly related to the node being processed, it provides a list where related nodes are placed together. By specifying directions on the edges, the DFT is restricted in exploring only the children of the node being processed. When a node has multiple parents and has a stronger connection to one of its unexplored parents than any of its children (like node G in Fig. 4), it would be obviously better to place that parent node next to the node being processed.

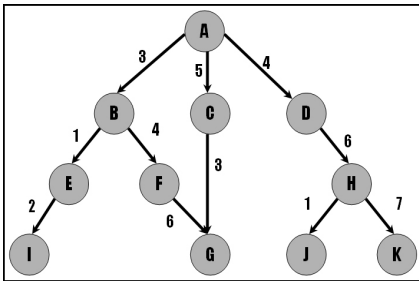


Fig. 4. Traversal Example

To facilitate this, the *Adjacency Matrix* of the parameter tree is symmetrized. This allows the DFT to move to any node connected to the node being processed. This technique is called *Bi-Directional Depth First Traversal* (BDFT). Finally, a *Depth Limited Traversal* would ensure that a single branch does not monopolize the optimal placement condition.

Thus the path taken by the DFT algorithm on this graph is $A \rightarrow C \rightarrow G \rightarrow D \rightarrow H \rightarrow K \rightarrow J \rightarrow B \rightarrow F \rightarrow I \rightarrow E$ while path taken by BDFT is $A \rightarrow C \rightarrow G \rightarrow F \rightarrow B \rightarrow D \rightarrow H \rightarrow K \rightarrow J \rightarrow E \rightarrow I$. To estimate the placement in the list, a metric known as *Placement Factor* is computed. The placement factor gives a measure of how well a node is placed in the list with respect to its distance from related nodes. Obviously,

a uniform placement factor for all nodes is desirable. This would mean that every node is equally well placed in the list.

The equation of placement factor of Node i is given by:

$$\text{Placement Factor (Node } i) = \sum (W_{ij} \times D_{ij}) / \sum (W_{ij}) . \quad (1)$$

with summation over j and where W_{ij} is the weight between Node i and Node j and D_{ij} is the distance between Node i and Node j in the list.

Smaller values of the placement factor indicate that the placement of that particular node is good. Initially the average placement factor is computed. Then all nodes having values greater than the average are arranged so as to reduce their placement factor value. This arrangement is done one node at a time starting from the worst placed node. When the placement factor of that node is reduced below the average, the placement factors for all nodes are recomputed and the entire process repeated. To avoid oscillations, a history of changes is kept. Any recurring pattern is identified and the responsible nodes are blacklisted. Thus the overall effectiveness of placement of the nodes in the list is improved.

4 Encoding

The process of creation of the actual DNA strands is carried out in the encoding phase. A DNA sequence that totally describes a processor on decoding is bound to be complicated. Such complicated sequences will be difficult to work with during recombinations unless they are well formed and ordered. Thus a good encoding scheme must be biologically realistic, yet well ordered. By placing related nodes near each other, we allow *Localization of Reactions* wherein related characteristics are changed simultaneously. This bio inspired node placement scheme provides a good platform to build simple recombination rules. The encoding process transforms information to a base 4 domain. The base 4 system allows us to define nature inspired recombination rules. Moreover, if characteristics of a species can be related to each parameter of the processor, then the DNA strands defining those traits can be used. By using this trait based encoding and natural DNA recombination rules, evolution can be completely realistic.

The entire DNA strand for a particular processor is split into 2 parts. The *Active Component* participates in the recombination. This component represents the architectural details as well as the *Instruction Set* details. It is created in the encoding stage and expanded in the recombination stage i.e., the basic architectural details such as 32 bit instruction length is added to the DNA string in the encoding stage while finer aspects such as the usage of a *Carry Save Adder* (CSA) is added to the string during recombination stage. The *Passive Component* is formed in the mutation stage. It consists of the *Finite State Machine* (FSM) description, the *Netlist* and basic *Placement* details of the processor. The netlist defines the connectivity across various components of an electronic design while placement is the process of assigning exact locations to various components in the chip's core area. The active components of various processors react with each other during recombination while the passive components are formed during mutation. They do not take part actively during recombination. But, the FSM details of the two processors are inherited by the

offspring based on its instruction set. These inherited FSMs are then used as guidelines to form the actual FSM of the offspring.

Variable Length Delimiter Based Encoding (VLDBE) system is used for encoding. A delimiter is a special sequence which serves as a flag that represents the beginning of a node, a field, a number, a character, a symbol or any other data entity. The delimiter is set as ATCG** where * is a wild card that can be substituted by any of the symbols. Each substitution signifies a different delimiter. In particular, AA signifies **No Delimiter**. This is very important to identify a data sequence which contains a substring 'ATCG'. For example if a data contains the string GAGCTATCGCGA, then this sequence would be transformed and encoded as GAGCTATCGAACGA. The 'AA' string signifies that the ATCG sequence was a data element rather than a part of a delimiter sequence. The 'AA' string is ignored when the sequence is decoded. Each symbol in a DNA sequence is known as a *Dit*, which is an acronym for a DNA digit. Fig. 5 lists various delimiters and their associated sequences.

| SEQUENCE | MEANING | SEQUENCE | MEANING |
|----------|--------------------|----------|-------------------|
| ATCGAA | NO DELIMITER | ATCGCA | CHILD LIST BEGINS |
| ATCGAT | NODE BEGINS | ATCGCT | NUMBER BEGINS |
| ATCGAC | DATA FIELD BEGINS | ATCGCC | NETLIST BEGINS |
| ATCGAG | ALHPASTRING BEGINS | ATCGCG | FSM DETAILS BEGIN |
| ATCGTA | NODE FUNC. BEGINS | | |
| ATCGTT | NODE ID BEGINS | ATCGG* | SPECIAL ESCAPE |
| ATCGTC | NODE NAME BEGINS | | SEQUENCE RESERVED |
| ATCGTG | PARENT LIST BEGINS | | FOR FUTURE USE |

Fig. 5. Table of Delimiters

As an example, a straight forward encoding scheme for number is discussed here. This encoding scheme for numbers involves a straight forward conversion to base 4 system.

The mapping of A, T, C and G is $A \rightarrow 0$ $T \rightarrow 1$ $C \rightarrow 2$ $G \rightarrow 3$. The numbers themselves are represented in the format shown in Fig. 6. The first field, the **Number Delimiter** is used to identify the sequence as a number bearing entity.

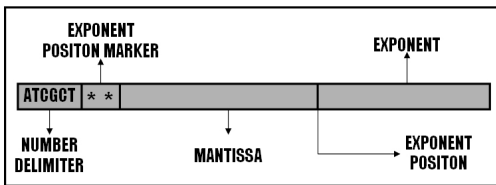


Fig. 6. Number Format

The next field is the **Exponent Position Marker (EPM)** which a 2 dit field. It shows where the **Mantissa** field ends and where the **Exponent** begins. The first dit of the mantissa decides its sign. If the first dit is A, then the mantissa is positive while if the first dit is G then the mantissa is negative.

If the dit is T or C, then the string has no meaning. Similarly, the first dit of the exponent decides its sign. The mantissa can have a maximum size of 16 (EPM = GG).The exponent on the other hand can be of any size. It ends when the next delimiter is encountered. Thus numbers of any magnitude can be represented with a precision of up to 16 decimal places.

5 The Gene Pools

There are two different gene pools: the **Processor Gene Pool** (PGP) and the **Heuristic Gene Pool** (HGP). The processor gene pool stores the complete design of various processors as well as several different functional units while the heuristic gene pool stores various heuristics used in the recombination and their respective effectiveness value. Architectural details, instruction set and FSM design of a microprocessor are stored as DNA sequences in the PGP. The processor strings are stored in an indexed array format for ease of retrieval. Each processor has an associated **Potency Factor**. The potency factor is a measure of a processor's capability in producing an offspring with a specific characteristic (metrics). This is not a simple numerical value, rather it is a vector. The vector (of n metrics) stores the potency of the processor to produce various types of processors. The n metrics are decided by the user based on his opinion of a processor's goodness. For example, a user might want to use power and performance as two important metrics while another user may use chip area, which ultimately decides cost of a processor, as an important metric.

Apart from this, various components of a processor such as different types of adders, multipliers, cache structures, micro control units etc. are stored in the gene pool. These are pre-encoded by the user i.e., the exact modular structure of these units and other details are stored as DNA sequences by manually forming these strings before starting the methodology. The details that are mentioned include dependency of the module with other components, modular placement and routing details, power and performance characteristics etc. These components can be chosen from the pool based on user specification. New modules can be added to the gene pool in the specific format as and when they are created. The heuristic gene pool on the other hand stores the various heuristics employed during recombination along with their **Degree of Belief**. The degree of belief is the amount of trust the methodology has on that particular heuristic. The heuristics are of two categories: **Recombination heuristics** and **Mutation heuristics**. The recombination heuristics are operations that can be done on the two reacting sequences during recombination. These may be natural operations such as splicing or arithmetic and Boolean operations. Mutation heuristics are algorithms to optimize the given architecture and also to generate the netlist along with the FSM.

6 Recombination

The microprocessor DNA sequences are combined to form an offspring. The recombination algorithm relies on the principle of **Localization of Reaction** according to which the probability of a node participating in recombination is determined by its proximity to a reacting node. In simpler terms, nodes react in bunches. This nature inspired scheme makes sense as adjacent nodes are strongly related. To achieve this, a function known as **Spread of Recombination** (SoR) is defined. The SoR can be any strictly decreasing function. It gives the **Probability of Recombination** of each node based its distance from the currently reacting node.

DNA strings can combine in many ways. In this methodology, two types of combinations are discussed. The **Dit-Wise Recombination** involves two strings

reacting dit by dit. Moreover, heuristics mentioned in the next section are applied to a bunch of adjacent dits. There is no sense of demarcation of nodes, fields etc. This technique is biologically realistic and very random. The biggest advantage of this method is the evolution of radical offspring. On the other hand this technique may produce large number of invalid strings before providing the required architecture.

The **Node Recombination** involves 2 strings which react node by node. The nodes can be identified using their delimiters. Nodes adjacent to the node currently undergoing recombination also react (with a probability). Two distinct types of field recombinations are possible. **Homogeneous Recombination** takes places between 2 similar nodes. Since these nodes represent the same characteristic of the processor, only the values of these nodes are affected. **Heterogeneous Recombination** takes place between 2 dissimilar nodes. Since the nodes do not represent the same characteristic of the processor, the recombination between them results in either the loss of information of a node or the formation of a new node. In the former case, the node that is lost is reformed afterwards in the mutation stage and in the latter case the newly created node is stored as a special sequence. During the decoding phase, the user is informed about this new node. The user can then either discard this new node and revert back to the 2 nodes which combined to create them or incorporate this new node in the design of the machine.

Thus, in this manner, yet to be discovered architectural features can be evolved. For example, suppose 2 cache-less processors react with each other and the **Latch Size** node of the first processor combines with the **Main Memory Size** node of the second processor, the resultant node can be interpreted as a node bearing the information about size of the cache. Thus the concept called cache is evolved from processors which did not possess them in the first place. Thus the methodology can discover new architectural paradigms through simple combinations of existing concepts. The heterogeneous recombinations occur with a slightly smaller probability when compared to homogeneous recombinations. Moreover, even in the event of heterogeneous recombination, the probability of discarding a node and reforming it at a later stage is higher than the probability of formation of a new node. The former probability is termed as **RFactor1** and the latter is termed as **RFactor2**. These parameters can be modified by the user. By default, these are set to 0.33 and 0.66. $RFactor1 \times (1 - RFactor2)$ gives the overall probability of a new node evolving. By default this value is 0.15, which means that 3 in every 20 combinations will result in a new node. Nodes formed after recombinations are immediately validated. For example, the **Number of Cache Levels** can only be a natural numbers.

7 Heuristics Update Strategies

The basic heuristics involved during recombination vary from natural splicing to Boolean and binary operations. These are stored in the **Heuristic Gene Pool** (HGP) along with their associated **Degree of Belief** (DoB). The heuristics are selected as per their degree of belief; higher the DoB, greater is the probability for that heuristic being used for recombination.

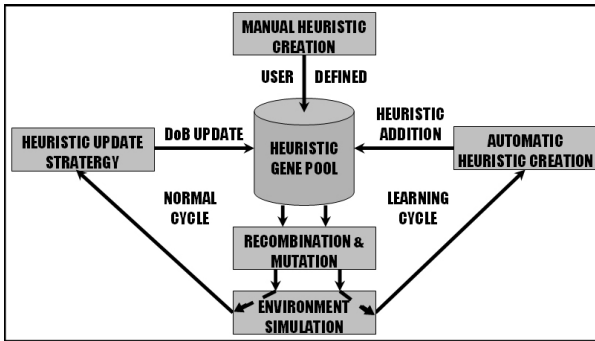


Fig. 7. Heuristic Life Cycle

Initially, all DoB are set to a default value of 0.66. The DoBs are updated once every H design cycles. Fig. 7 shows the heuristic life cycle. The basic heuristics are created beforehand. Also, user defined heuristics can be added to the HGP. Apart from these, *Automatic Heuristic Creation* in the *Learning Cycle* adds new heuristics to the HGP.

The learning cycle slightly modifies a heuristics present in the HGP and uses that for recombination. Over several iterations, various modifications of the heuristics are evaluated. If a modification yields better results consistently, then this new heuristic is added to the HGP. It is to be noted that the original heuristic is not modified; its modified version is added the HGP. In the normal cycle, each heuristic involved to evolve that offspring is noted. Once the *Processor Evaluation* is complete in the *Environment Simulation* stage, the responsible heuristics are collected in a set. Thus each iteration yields a specific set.

The *Heuristic Update Rate* (H) is a parameter which decides the number of iterations before which the heuristics are updated. The DoB for each heuristic is obtained. The DoB value of a set is a global DoB value assigned to all elements of the set. All DoB values are relative. The most successful processor's set gets a DoB of 1. The other sets get a relative DoB value. Next, the intersection of H sets (S_{inter}) is taken and the DoB values of each heuristic present in S_{inter} is computed by averaging the DoB values of each set in which they were originally present. At this stage various heuristics have relative DoBs ranging from 0 to 1. Next, each relative DoB is reduced by 0.5 making their range from -0.5 to 0.5. Finally, these Reduced DoBs are normalized by dividing it by the *Normalizing Factor* (N) which is by default set to 10. These normalized DoB values are then added to the DoB values of the respective heuristics in the HGP. Normalization is done to ensure that the DoB values do not oscillate wildly.

8 Creating DNA Mutation Algorithms

Mutation in this methodology refers to the formation of DNA strings that represent the netlist and FSM information. Once the architectural details of a processor are formed, the required functional units are fetched from the PGP. Once all the functional units are assembled, their interconnections are determined and a logical connectivity matrix is formed. Netlist generation and initial placement is carried out in the DNA domain. For this effect DNA based algorithms need to be devised. Conventional CAD algorithms for placement and routing can not be directly applied without decoding the processor string. Instead, if the DNA counter parts of these

algorithms are devised, then these processes can be carried out in the DNA domain itself. Apart from perfectly suiting a DNA computational model, aspects of developmental biology can be incorporated in the interconnect development during the decoding phase. The following section describes a novel methodology for creating such DNA algorithms from conventional algorithms. Fig. 8 illustrates this methodology.

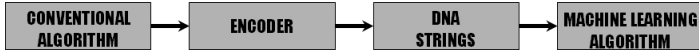


Fig. 8. Technique for Creating DNA Based Algorithms

The problem parameters at each of each step of a *Conventional Algorithm* is tracked and converted into a DNA string using an *Encoder*. Thus, for every step, a set of DNA sequences are obtained. A *Machine Learning Algorithm* draws inferences from the changes in the problem parameters after each step and creates a corresponding DNA action step. These action steps constitute the DNA algorithm. Several examples are given to the system to enable it to learn.

9 Decoding, Optimization and Environment Simulation

The offspring DNA obtained after the mutation process contains information about the connectivity of the modules, the netlist and the FSM of the offspring processor. Though the netlist is generated during the mutation phase, its placement is not optimized completely. This phase involves decoding the offspring DNA sequence to the conventional domain and optimizing the placement of the modules and the netlist.

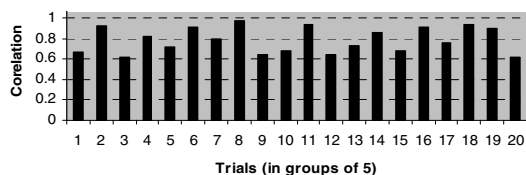
The offspring DNA sequence is read till a delimiter is encountered. The sequence between two node delimiters is interpreted as a node and is added to the parameter tree. The sequence inside the node is further decoded with the help of the delimiters to obtain its dependencies. In this manner the parameter tree is constructed. When an unknown node is encountered (if the node name does not match with a stored list of node names), the node is stored separately for further investigation. The details of the connectivity of the modules are also extracted from the offspring DNA sequence. The main advantage of evolution is that it is need based. The environment of a species plays a major role in the selection process and also may cause mutation of the DNA to make the species better suited to their environment. In the context of microprocessor design automation, environment simulation involves simulating the working environment of the microprocessor. The working environment of a processor is defined as the set of applications that are to be run on it. A generalized benchmark, for both ASIC as well as General Purpose Processor, is under development.

The algorithms that are to be executed on the processor must be converted to the processor's instruction set. A *Generic Compiler* that takes in the processor's instruction set as an input and translates the *Benchmarking Algorithms* in terms of the processor's instructions is designed for this purpose. Based on the processor's architecture, the delay and the power consumed by each instruction are computed. *Processor Simulator* simulates the working of the processor by simulating the execution time and the power consumed by each instruction. The benchmark

algorithms are virtually executed on the processor. Based on certain pre-defined heuristics, the processor characteristics are slightly varied to suit the algorithm's need. A simple example would be increasing the cache size if the algorithm involves large number of repeating instructions.

10 Simulations and Results

To test the methodology, a simple simulation was carried out. For this purpose 38 characteristics of 3 Intel processors (8085, Pentium, Pentium 4) were collected. Then their respective parameter trees were built and their DNA sequences were created using the VLDBE system. Then 8085 and Pentium 4 were recombined using the node recombination heuristics. The offsprings produced were validated. The parameter tree of valid offsprings were then correlated with the parameter tree of Pentium. The best correlation obtained every 5 trails was plotted. The maximum correlation was found to 0.97 (37 out of 38 parameters had the same value). These results are shown in Fig. 9. The simulation showed that some of the offspring of 8085 and Pentium 4 was similar to the Pentium processor.



The significance of this result is that it proves that it is possible to evolve the architectural design of a microprocessor using the methodology.

Fig. 9. The Correlation Graph

11 Generalization to Other Domains

Based on the microprocessor design methodology, a general scheme to use DNA based evolution for solving design automation problems is evolved. In *The Problem Domain*, the problem's data is represented as such. For example, consider the design automation of an aircraft. The various parameters of an aircraft can be represented by the tree structure. These would include wingspan, no. of tail rudders, service ceiling, wing loading etc. Relationships between these parameters are found and suitable node functions are created. *Arbitrary/ Existing Solution Creation* would involve creating the tree for existing aircrafts. The data structure can then be encoded into DNA sequences in a biologically realistic way. After *Encoding*, the solutions are now present in the *DNA Domain* where they undergo biologically realistic recombination and mutation. These reactions can be stochastic in nature guided by a few domain specific heuristics. For example the output of the homogenous combination of 2 wing structures will depend on the type of fuselage. *Mutation* heuristics can be learnt and these, along with recombination heuristics, can be refined using heuristic update strategies. DNA based *Post-Processing* is carried out to rectify minor flaws in what is otherwise a good solution. For example, if a cargo aircraft is evolved with 4 wheel landing gear, then the number of wheels can be increased based on its capacity.

Solution evaluation involves environment (physical) simulations. The metrics for aircraft design evaluation could be Rate of Climb, Thrust to Weight Ratio etc. Post processing might employ conventional optimization techniques like simulated annealing, game theory, genetic algorithms etc. For example, simulated annealing can be employed to optimize the wing area. Good solutions are selectively added the **Gene Pool**. These genes are used during the recombination phase. Moreover, the heuristics employed to obtain the current solution is also found out. The values for the goodness of the heuristics are then updated in the **Heuristic Gene Pool**. Over several iterations, the gene pool is enriched. Both the heuristics used as well as the quality of solutions obtained is improved. After many such iterations, the final desired solution can be obtained.

12 Conclusion

This paper presented a novel methodology for naturally evolving microprocessors as per user specifications. This is only the beginning of a new vista in design automation. The encoding process can be made more biologically realistic using trait mapping techniques or amino acid based encoding schemes. Control logic synthesis can be incorporated as a part of the DNA domain design automation process. Once the DNA based design automation for microprocessor stabilizes, the turn-around time would reduce drastically. Further, this methodology can be extended to other domains such as aircraft and automobile design.

References

1. N Venkateswaran et al: Microprocessor Design Automation: A DNA Based Evolutionary Approach. BICS 2006.
2. David B. Fogel: Evolutionary Computation: The Fossil Record. Wiley-IEEE Press (May 1, 1998)

Multiple Sequence Alignment with Evolutionary-Progressive Method

Paweł Kupis and Jacek Mańdziuk

Faculty of Mathematics and Information Science,
Warsaw University of Technology,
Plac Politechniki 1, 00-661, Warsaw, Poland
kupisp@student.mini.pw.edu.pl, mandziuk@mini.pw.edu.pl

Abstract. A new evolutionary-progressive method for Multiple Sequence Alignment problem is proposed. The method efficiently combines flexibility of evolutionary approach with speed and accuracy of progressive technique. The results show that the hybrid method is an interesting alternative for purely genetic or purely progressive approaches.

1 Introduction

Multiple sequence alignment (MSA) is one of the most important tasks in computational biology. The problem is NP-Hard [1] and consequently high computational complexity and memory requirements make it hard to be approached by the exact, dynamic programming methods (e.g. [2]). In practice dynamic programming methods could be accepted as an effective tool only for pairwise sequence alignment (PSA). In the true MSA case (i.e. for $n \gg 2$, where n denotes the number of sequences to be aligned) their computational load is prohibitive and instead alternative approaches are usually exploited at the cost, however, of losing the guarantee of finding the optimal solution.

The main alternative to dynamic programming methods is *progressive method* [3,4], which relies on a series of pairwise alignments in order to build up a final alignment. Closely related sequences are aligned first and subsequently more distant ones. Progressive methods differ in the way the pairwise sequence distance matrix is calculated which has immediate impact on the order according to which sequences are added to the partial solution maintained by the method. In the most renown progressive approach - Clustal W [3] (and its various refinements e.g. [5]) the alignment order is determined by the *phylogenetic tree*, which defines evolutionary distance between sequences. Despite the greedy nature (which can be partly alleviated [6]) the method, due to its high speed and reasonable accuracy, remains one of the most popular tools for solving MSA problem.

In this paper, following [7], we present a hybrid evolutionary-progressive (E-P) method for simultaneous aligning of several amino acid sequences. Due to the space limit several introductory notions concerning MSA as well as foundations of *genetic algorithms* (GA) / *evolutionary programming* (EP) are omitted in the paper. For examples of GA/EP-based approaches to MSA please refer e.g. to [8,9,10,11].

2 Evolutionary-Progressive Method

In the straightforward EP-based approach to MSA each individual in a population represents an alignment. Despite its simplicity, such representation suffers from a very large search space and consequently dramatically increases the execution time. Alternative idea is to apply EP to obtain initial, partial alignment in the restricted search space and subsequently use another method to achieve the final solution.

One of the promising examples of such hybrid approach was presented by Zhang and Wong in [7]. In the first step evolutionary algorithm is used to find the first approximation of the final alignment (called pre-alignment) and then the aligned columns are fixed. In the next step the elements between the pre-aligned columns are aligned by a greedy algorithm using pairwise alignment. The method looks promising, but since several relevant implementation details are missing in [7] it is not possible to exactly follow the idea. In particular the question about how to build the initial population in the pre-alignment space (which is crucial for the quality of obtained result) is not addressed. Another question that can be raised concerns the usefulness of genetic operators proposed in [7]. Our claim is that mutation operator of the form presented in [7] is inefficient. Additionally some refinements are proposed to the definition of a crossover operator. Finally, in the second phase of the algorithm we propose to use the progressive method.

In the next two subsections our modified hybrid method is presented in more detail followed by experimental results (Section 3) and conclusions (Section 4).

2.1 The Evolutionary Stage

A definition of pre-alignment uses the notions of *identical column* and *columns block*. Column of alignment is called identical if its elements (symbols in each row) are all the same. Identical columns form a block if they are neighbors in alignment. An example presented below shows three sequences, all possible identical columns that can be defined and possible formed blocks. Numbers in columns are indices of respective sequences.

| | | | | | | | | | | | | | | |
|----------|---|---|---|---|---|---|---|---|---|---|---|---|-----|-----|
| MAAFCP | 1 | 1 | 1 | 1 | 2 | 3 | 4 | 5 | 5 | 5 | 5 | 6 | 1 2 | 5 6 |
| MACFMCPC | 1 | 1 | 5 | 5 | 2 | 2 | 4 | 3 | 3 | 6 | 6 | 7 | 1 2 | 6 7 |
| MACMFCPC | 1 | 4 | 1 | 4 | 2 | 2 | 5 | 3 | 6 | 3 | 6 | 7 | 1 2 | 6 7 |

Pre-alignment is defined as a series of identical column blocks which fulfils the following conditions:

- in each row, each number (index) can appear only once,
- in each row numbers are in ascending order.

Each single column is treated as a block of length one. The above conditions guarantee that the final alignment can be build on the basis of pre-alignment.

In our approach, similarly to Zhang and Wong method, an individual in evolutionary algorithm represents a pre-alignment as defined above. The search space is restricted to all correct pre-alignments of a given set of sequences. The first

problem to solve is generating the first population of individuals (i.e. the initial set of pre-alignments). Clearly, identification of all possible identical columns is inefficient, since the complexity of this task equals the complexity of the whole MSA problem. Moreover, a number of columns found would be too big to generate efficient population. For generating the first population and for the whole evolutionary process the notion of *harmful block* is considered. A formal definition of a harmful block and measure of its harmfulness can be found in [7]. Intuitively a harmful block can be described as the one connecting two too distant parts of sequences (see Fig. 1). The method of identifying possible identical



Fig. 1. Schematic example of harmful a block

columns should not prefer such harmful columns, but on the other hand it ought to utilize all symbols in sequences to build a representative subset of identical columns. At last upper limit of the columns found and execution time should be restricted to reasonable limits. After several preliminary trials the following method, depicted in Fig. 2, has been developed. The method is characterized

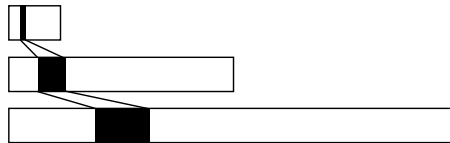


Fig. 2. The idea of search windows

by two parameters - c_{max} is the upper limit of the columns that compose the pre-alignments and $w_{\%}$ defines the size of a search window (which equals $w_{\%}$ of the sequence length).

At first, one of the sequences, denoted by S , is chosen (in our implementation it is always the shortest one) and then the following procedure is repeated some predefined number of times: a symbol $s \in S$ is selected and its relative position (denoted by rp) with respect to the beginning and the length of S is determined. Next, for each sequence other than S a window of width $w_{\%}$ of the length of sequence is defined and its midpoint is positioned at rp . Then, in each

¹ Please note, that in [7] a harmful block is used for finding the sub-optimal mutation points during the EP, and not for generating the initial population. Here we suggest to include its notion already in the process of defining the initial population, and to skip the mutation operator.

sequence one symbol within window's range is randomly selected. If all selected symbols are pairwise equal to s , then their indices create a new identical column. Assuming the uniform distribution of identical columns in S , the above procedure is repeated $\lceil \frac{c_{max}}{|S|} \rceil$ times for each symbol $s \in S$. Symbols in S are selected in ascending order of their indices. The created identical columns are stored in order of creation. Finally, the initial population of pre-alignments is generated using the following procedure (c_p is the population size, A is an ordered set of identical columns found with the above described method and P is a set of pre-alignments; initially P is empty)

```

foreach( $a$  in  $A$ ) {
  foreach( $p$  in  $P$ ) {
    if( $a$  could be added at the end of  $P$ ) {
      join  $a$  to  $P$ ;
      if possible
        join together  $a$  and the last block in  $P$ ;
      goto next  $a$ ;
    }
  }
  create new  $p$  using  $a$ ;
  join  $p$  to  $P$ ;
}
sort  $P$  by fitness function value;
choose no more than  $c_p$  best individuals;

```

The above procedure gathers information about identical columns in restricted number of individuals. It does not guarantee optimal usage of the columns found, but if the order of columns is preserved from the search operation, results are acceptable against execution time. The default values used in our method are $c_{max} = 4\,000$, $w\% = 0.04$ (4%) and $c_p = \frac{m_a \times n}{10}$, where m_a is the mean sequence length and n is the number of sequences. c_p is additionally restricted to the interval $< 100, 400 >$. Please note, that even if the initial population size is smaller than c_p it will be enlarged to c_p by the first selection operation.

Once the first population is generated the evolutionary process begins. The method uses traditional selection operator - fitness proportionate selection, also known as roulette-wheel selection with one modification - the best individual is extra promoted. Originally mutation operator was planned to be implement as described in [7], but preliminary tests revealed that it was very hard to set the threshold in order to eliminate harmful blocks automatically. Since the major function of mutation is to prevent formation of too long alignments it was decided to control this by appropriate fitness function instead. Consequently mutation operator is not used in the proposed method. The proposed function is defined as follows (p is an individual):

$$fitness(p) = 100 \times \frac{col(p)}{(len_{min}(p))^\alpha} \quad (1)$$

where $col(p)$ returns the number of columns in p and $len_{min}(p)$ returns the minimal possible length of alignment constructed on the basis of pre-alignment represented by individual p . More precisely, let the i -th block of pre-alignment p be denoted by b_i :

$$\begin{array}{cccc}
b_{i_{1,1}} & b_{i_{2,1}} & \cdots & b_{i_{w_i,1}} \\
b_{i_{1,2}} & b_{i_{2,2}} & \cdots & b_{i_{w_i,2}} \\
\vdots & \vdots & \ddots & \vdots \\
b_{i_{1,n}} & b_{i_{2,n}} & \cdots & b_{i_{w_i,n}}
\end{array}$$

where w_i is the width of b_i and let m denotes the number of blocks in pre-alignment p and s_r the length of the r -th sequence, then:

$$len_{min}(p) = \max_{1 \leq j \leq n} (b_{1,j}) + \sum_{k=2}^m \max_{1 \leq j \leq n} (b_{k,j} - b_{k-1,j}) + \max_{1 \leq j \leq n} (s_j - b_{m,j}) \quad (2)$$

Exponent α in [\(1\)](#) specifies the significance of *length penalty* ($\alpha = 20$ by default).

In the crossover operation a random cutting point for each of the two chosen pre-alignments is independently selected and subsequently individuals exchange information. A cutting point never splits existing blocks. Additionally, crossover operator merges blocks in the offsprings if possible (it is not a costly operation, since only blocks neighboring the cutting points have to be checked). Another modification is adding a condition that preserves the best individual in the population, i.e. at least one of the children has to be better than both of the parents in order to allow children replace their parents. Also incorrect pre-alignment never replaces its parents. Cutting point before the first or after the last block causes empty pre-alignment, that alignment also never replaces its parents. Crossover probability was set to 0.4. Evolutionary process can be stopped due to one of the following reasons:

- fitness of the best individual did not change in the last 40 generations,
- the limit of 1 000 generations was exceeded.

After termination of the evolutionary algorithm the best individual is selected and the evolutionary method is recurrently called for substrings located between its blocks. Recursion is stopped if at least one of the following conditions is fulfilled:

- the maximum distance between neighboring blocks is less than 20,
- the algorithm found no identical columns between neighboring blocks.

In that case the progressive method ([Section 2.2](#)) is called for the remaining substrings.

2.2 The Progressive Stage

In the second stage a typical progressive algorithm is used. In our implementation it is Clustal W [\[3\]](#) like method. A phylogenetic tree is built with the use of neighbor-joining and mid-point rooting methods. For pairwise alignment Myers-Miller method [\[12\]](#) is applied enhanced to use position-specific gap penalties and other improvements described in [\[3\]](#), in particular:

- sequence weighting,
- gap opening penalty (GOP) modification depending on existing gaps,

- gap extension penalty (GEP) modification depending on existing gaps,
- GEP modification depending on difference in the lengths of the sequences.

Please note that implementation of the progressive part of the whole MSA method must also be very efficient because in some cases only a few or even no identical columns could be found.

3 Results

We compared our implementation of evolutionary-progressive method with ones of the most popular programs in both evolutionary and progressive category. We chose SAGA 0.95 [13] and Clustal W 1.83 [14] as representatives of evolutionary method and progressive one, resp.

Two different measures were used to compare quality of the produced alignments. The first one is the sum-of-pairs score (SPS) and the second one is the column score (CS). Pairwise alignment score for SPS is calculated exactly the in same way as in dynamic programming method and includes substitution matrix usage and affine gap penalty. Two SPS are calculated for the following two parameter sets and finally the mean value is calculated:

- *Set 1* – GOP: 10, GEP: 0.2, BLOSUM 62 matrix,
- *Set 2* – GOP: 10, GEP: 0.2, 250 PAM matrix.

SPS represents the cost of alignment, which means that *the lower SPS, the higher quality of alignment*. CS is calculated in a standard way defined in [15]. On the contrary to SPS for the CS measure *the higher its value the better the alignment result*. Both measures are defined to have nonnegative values. Quality of alignment produced by any of the programs is measured in relation to quality of the reference alignment. Thus, values truly used in comparisons are related to measures value for reference alignments and expressed in percentage.

Test cases were taken from the reference database - BALiBASE, which is the most widely used multiple alignment benchmark, providing high quality, manually refined, reference alignments. We used version 2.01 of that database and also the latest release 3.0. Version 3.0 of BALiBASE [16] includes new, more challenging test cases, representing the real problems encountered when aligning large sets of complex sequences.

All tests were executed on desktop PC (AMD Athlon XP 2000+ 1.70 GHz with 1.00 GB memory). Clustal W 1.83 and our method were run under the control of MS Windows Server 2003, SAGA 0.95 worked under Fedora Core 3 Linux. E-P method was implemented in C# 2.0. Default parameter settings were used in all programs. SAGA used MSA objective function. A single test case was marked as successfully completed if execution time was no longer than 1 hour and memory consumption did not exceed 1 GB. First, test cases from the version 2.01 were taken. The results are presented in Table 1, where it is shown that only SAGA did not complete successfully all test cases. In almost 10% of test cases at least one of test limits was exceeded. Also, it can be seen that our E-P method is a little faster than Clustal W and significantly faster than its evolutionary competitor SAGA.

Table 1. Results obtained for BALiBASE ver. 2.01 test cases

| | the average SPS | the average CS | summary execution time | the length of alignment | successfully completed test cases |
|-----------|-----------------|----------------|------------------------|-------------------------|-----------------------------------|
| SAGA | 101.9 | 77.1 | 51503 | 94.6 | 90.8 |
| Clustal W | 101.2 | 89.7 | 90 | 96.6 | 100 |
| E-P | 105.5 | 92.7 | 38 | 102.0 | 100 |

The SPS values comparison shows that SAGA and Clustal W obtained very similar results. The cost of alignments produced by E-P method was a bit higher. As follows from comparison of the CS values the E-P method obtained topmost result in this category. Again the scores of SAGA and Clustal W are comparable.

Based on the above results it was decided to use test cases from the newer version of the database only with programs which successfully completed all tests from version 2.01. The results obtained for BALiBASE 3.0 are presented in Table 2 (all tests were successfully completed). The general conclusion from

Table 2. Results obtained for BALiBASE ver. 3.0 test cases

| | the average SPS | the average CS | summary execution time | the length of alignment |
|-------------------------|-----------------|----------------|------------------------|-------------------------|
| Clustal W | 103.6 | 64.9 | 2902 | 94.3 |
| E-P | 104.6 | 74.2 | 1492 | 97.3 |
| E-P ($w_{\%} = 0.08$) | 104.0 | 95.0 | 902 | 98.9 |
| E-P ($w_{\%} = 0.12$) | 102.6 | 105.6 | 825 | 101.2 |
| E-P ($w_{\%} = 0.16$) | 101.8 | 119.8 | 795 | 103.1 |
| E-P ($w_{\%} = 0.20$) | 100.5 | 123.1 | 768 | 104.8 |
| E-P ($w_{\%} = 0.24$) | 99.6 | 126.5 | 757 | 107.7 |
| E-P ($w_{\%} = 0.28$) | 98.7 | 134.2 | 777 | 108.7 |

Table 2 is that E-P method is comparable in both time and quality with Clustal W. Actually, by adjusting the window's width $w_{\%}$ in the evolutionary process one can easily establish the balance between the quality measures (SPS, CS) and the execution time or alignment's length. For example E-P excels Clustal W in the SPS category for $w_{\%} \geq 0.12$. Also for all tested window's lengths the proposed method outperforms its competitor in the CS measure and the execution time. On the other hand, regardless the choice of $w_{\%}$ the length of alignment output by E-P method exceeds the one yielded by Clustal W.

4 Conclusions

The efficient MSA method can be build using evolutionary techniques, but the representation of individuals and definition of the search space have to be suitably mated. Traditional way of representation is impractical. The evolutionary-progressive method described in this paper is a compromise between flexibility

of evolutionary approach and advantages of progressive method, which are speed and quality. The new concept of search space definition makes evolutionary part of the algorithm easier to implement and faster. Progressive part is used for subtasks with reduced problem dimension. Combination of these two ideas creates the method which is an interesting alternative for progressive methods and which is competitive to purely evolutionary approaches.

References

1. Wang, L., Jiang, T.: On the complexity of multiple sequence alignment. *Journal of Computational Biology* **1(4)** (1994) 337–348
2. Carrillo, H., Lipman, D.J.: The multiple sequence alignment problem in biology. *SIAM Journal on Applied Mathematics* **48(5)** (1988) 1073–1082
3. Thompson, J.D., Higgins, D.G., Gibson, T.J.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.* **22** (1994) 4673–4680
4. Zhu, M-J., Hu, G-W., Zheng, Q-L., Peng, H.: Multiple sequence alignment using minimum spanning tree, Proc. 4th International Conference on Machine Learning, Guangzhou (2005) 3352–3356
5. Chaichoompu, K., Kittitornkun, S., Tongsim, S.: MT-ClustalW: Multithreading Multiple Sequence Alignment, Proc. International Parallel and Distributed Processing Symposium (IPDPS) (2006)
6. Notredame, C., Higgins, D., Heringa, J.: T-coffee: A novel method for multiple sequence alignment. *Journal of Molecular Biology* **302** (2000) 205–217
7. Zhang, C., Wong, A.K.C.: A genetic algorithm for multiple molecular sequence alignment. *Computer Application in the Biosciences* **13(6)** (1997) 565–581
8. Thomsen, R., Fogel, G.B., Krink, T.: Improvement of Clustal-Derived Sequence Alignments with Evolutionary Algorithms, Proc. The 2003 Congress on Evolutionary Algorithms, vol. 1 (2003) 312–319
9. Liu, L., Huo, H., Wang, B.: Aligning multiple sequences by genetic algorithm, Proc. International Conference on Communications, Circuits and Systems (ICCCAS'04), vol. 2 (2004) 994–998
10. Zhang, G-Z., Huang D-S.: Aligning Multiple Protein Sequence by An Improved Genetic Algorithm, Proc. 2004 IEEE International Joint Conference on Neural Networks, vol. 2 (2004) 1179–1183
11. Abdesslem, L., Soham, M., Mohamed, B.: Multiple Sequence Alignment by Quantum Genetic Algorithm, Proc. International Parallel and Distributed Processing Symposium (IPDPS) (2006)
12. Myers, E.W., Miller, W.: Optimal alignments in linear space. *Bioinformatics* **4(1)** (1988) 11–17
13. Notredame, C., Higgins, D.G.: SAGA: sequence alignment by genetic algorithm. *Nucleic Acids Res.* **24(8)** (1996) 1515–1524
14. Chenna, R., et al.: Multiple sequence alignment with the Clustal series of programs. *Nucleic Acids Research* **31(13)** (2003) 3497–3500
15. Thompson, J.D., Plewniak, F., Poch, O.: A comprehensive comparison of multiple sequence alignment programs. *Nucleic. Acids. Res.* **27(13)** (1999) 2682–2690
16. Thompson, J.D., Koehl, P., Ripp, R., Poch, O.: BALiBASE 3.0: Latest Developments of the Multiple Sequence Alignment Benchmark. *PROTEINS: Structure, Function, and Bioinformatics* **61** (2005) 127–136

Optimal Design Centring Through a Hybrid Approach Based on Evolutionary Algorithms and Monte Carlo Simulation

Luis Pierluissi and Claudio M. Rocco S.

Universidad Central de Venezuela, Facultad de Ingeniería,
Apartado Postal 47937, Los Chaguaramos, Caracas, Venezuela
lpierluissi@hotmail.com, crocco@reacciun.ve

Abstract. In many situations a robust design could be expensive and decision-makers need to evaluate a design that is not robust, that is, a design with a probability of satisfying the design specifications (or yield) less than 100 %. In this paper we propose a procedure for centring a design that maximises the yield, given predefined component tolerances. The hybrid approach is based on the use of Evolutionary Algorithms, Interval Arithmetic and procedures to estimate the yield percentage. The effectiveness of the method is tested on a literature case. We compare the special evolutionary strategy (1+1) with a genetic algorithm and deterministic, statistical and interval-based procedures for yield estimation.

1 Introduction

The robustness of a design is defined as the maximum size of the deviation from this design that can be tolerated whereby the product still meets all requirements [1]. As an example, consider the temperature controller circuit [2] shown in Fig. 1. The performance function is:

$$R_{T-on} = \frac{R_1 R_2 (E_2 R_4 + E_1 R_3)}{R_3 (E_2 R_4 + E_2 R_2 - E_1 R_2)} .$$

We can evaluate, for example, what are the maximum possible deviations for each component in the Fig. 1 consistent with $2.90 \text{ k}\Omega \leq R_{T-on} \leq 3.10 \text{ k}\Omega$? That is, the goal is to assess the input parameters uncertainty, which maintains the output performance function within specified bounds.

Suppose the area shown in Fig. 2a is the feasible zone F for a generic design with variables R and R_a . Within the feasible zone any pair (R, R_a) satisfies the specifications. However an exact description of the Feasible Solution Set (FSS) (Fig. 2a) is in general not simple and approximate descriptions are often looked for, e.g.: the Maximum Outer Box (Fig. 2b) or the Minimum Internal Box (MIB) (Figs. 2c and 2d).

Several authors have studied the robust design problems under the perspective of Evolutionary Strategies [3]-[7]. In particular, in [7] the authors propose an indirect approach to obtain the MIB based on the optimisation of the MIB volume instead of a direct method based on mapping from the output into the input space. The approach

uses an Evolution Strategy technique to maximise the volume, while Interval Arithmetic is used as a checking technique that guarantees the complete feasibility of the design, i.e. a 100 % yield.

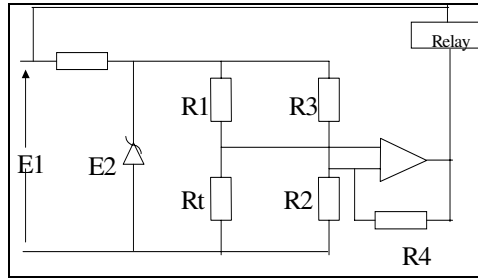


Fig. 1. Temperature Controller Circuit [2]

However there are situations in which design constraints (for example, predefined component tolerance) do not allow obtaining the MIB. So the decision-maker (DM) needs to propose a design with a probability of satisfying the design specifications (or yield) less than 100 %.

In this paper we propose a design approach that maximises the probability of satisfying the design specifications, given component tolerances predefined by a DM. The proposed approach combines an evolutionary approach during the optimisation phase, along with different procedures to estimate the yield percentage.

Section 2 contains an overview of Interval Arithmetic. The Evolutionary approach is presented in Sect. 3. Section 4 presents the general approach used to solve the yield

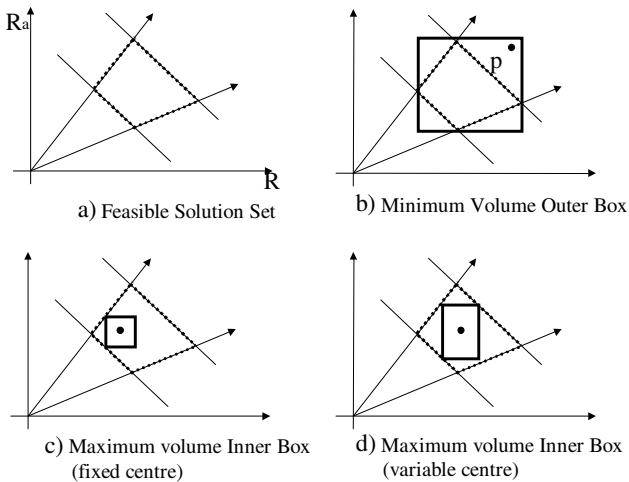


Fig. 2. Feasible Solution Set and approximate descriptions [3]

maximisation design. A real design evaluation is considered in Sect. 5, and finally Sect. 6 presents the conclusions.

2 Interval Arithmetic

Interval "numbers" are an ordered pair of real numbers representing the lower and upper bound of the parameter range. If we have two interval numbers $T=[a,b]$ and $W=[c,d]$ with $a \leq b$ and $c \leq d$ then $T+W=[a+c,b+d]$. Similar expressions can be defined for the other basic operations and for transcendental functions [8]-[10].

Only some of the algebraic laws valid for real numbers remain valid for intervals. An important property referred to as sub-distributivity does hold. It is given mathematically by the set inclusion relationship: $T(W+Z) \subseteq TW+TZ$. The failure of the distributive law often causes overestimation. In general when a variable occurs more than once in an interval computation it is treated as a different variable in each occurrence. This effect is called the "dependency problem".

Consider a real valued function f of real variables t_1, t_2, \dots, t_n and an interval function F of interval variables T_1, T_2, \dots, T_n . The interval function F is said to be an interval extension of f , if $F(t_1, t_2, \dots, t_n) = f(t_1, t_2, \dots, t_n)$. The range of a function f of real variables over an interval can be calculated from the interval extension F , changing t_i by T_i . Note that: $f(t_1, t_2, \dots, t_n) \subseteq F(T_1, T_2, \dots, T_n)$, for all $t_i \in T_i$ ($i=1, \dots, n$) [8].

3 Evolutionary Algorithms

Among the many heuristics from different fields that can be used to efficiently solve an optimisation problem we are interested in those that rely on analogies to natural evolution, the so-called Evolution Algorithms (EA). Genetic Algorithms (GA) and Evolution Strategies (ES) are among the most widely used EA and both have been successfully applied to a wide range of problems [11]-[13].

In this paper we compare the evolution strategy called (1+1) and GA. The choice of (1+1) has to do basically with the simplicity, noise tolerance and small population size it manages. The latter condition is of paramount importance for the problem at hand, since it reduces the number of quality evaluations of the solutions. As explained later, the quality of a solution requires the estimation of a time-consuming fitness function.

In the (1+1) technique each individual is represented by a couple $(\mathbf{X}, \boldsymbol{\sigma})$, where \mathbf{X} is a vector that identifies a point in the search space and $\boldsymbol{\sigma}$ is a vector of perturbations used to evolve the matching component of \mathbf{X} . In the next iteration an offspring is generated by the expression: $\mathbf{X}^{t+1} = \mathbf{X}^t + N(\mathbf{0}, \boldsymbol{\sigma}^2)$. In this way the solution at iteration t is perturbed and a new descendent for the following iteration $t+1$ is obtained. The offspring generated will be accepted as the new member of the population, only if it has a better quality than its parent. To improve the convergence rate of the (1+1), the "1/5 success rule" proposed in [13] is used.

On the other hand, GA are based on an initial populations of chromosomes (proposed solutions), with n individuals, that are combined to generate hopefully a new better population (i.e., with an increased mean fitness). This sequence continues until

a termination criterion is reached. As for the natural selection, the new population is generated by repeatedly performing the four fundamental operations of reproduction, crossover, replacement and mutation, all based on random sampling. [13].

For the GA implemented in this paper, we used:

A real-valued encoding, with constant population; Random initialisation; Linear Renormalisation; Standard Roulette wheel selection rule; Reproduction based on mutation and crossover operators; Linear interpolation for crossover and mutation probabilities; Uniform crossover; Partial substitution without duplicates.

Parameters: Population size =100, Crossover probability = 0.9, Mutation probability = 0.1, Gap = 5 %, Generations=100.

4 Centring Design Methodology

Given the tolerance of each component, the optimal centring design to be considered consists on selecting the co-ordinates of centre of the hyper-box that maximises the yield that is the probability of satisfying the design specifications.

Let B be the box defined by: $B := \{ \mathbf{x}, \mathbf{C} \in \mathbb{R}^n \mid x_i \in [C_i - \Delta x_i, C_i + \Delta x_i] \}$, where \mathbf{C} is the unknown centre co-ordinates vector, and Δx_i is the known tolerance for each variable.

The designer formulates target values on the quality of the product by setting lower and upper bounds on the property $y_i(\mathbf{x})$. We define the following slack function $g_i(\mathbf{x})$:

$g_i(\mathbf{x}) = UB_i - y_i(\mathbf{x})$ when there is an upper bound requirement or

$g_i(\mathbf{x}) = y_i(\mathbf{x}) - LB_i$ when there is a lower bound requirement

Let F be the set defined as $F = \{ \mathbf{x} \in \mathbb{R}^n \mid g_i(\mathbf{x}) \geq 0, i=1, \dots, m \}$. The idea is to obtain \mathbf{C} in such a way that the hyper-volume of $F \cap B$ is maximal, that is:

$$\max_{\mathbf{C}} \text{hyper-volume of } F \cap B$$

\mathbf{C}

$$\text{s.t. } \mathbf{C} \in F \cap B$$

Since the optimization problem is solved using an EA, a proper fitness function must be defined. In our case, the quality of a solution is measured by the objective function, i.e. the hyper-volume of $F \cap B$.

4.1 General Approach

Given r constraint functions $g_i(x_1, x_2, \dots, x_n)$, lower and upper bounds (LB_i, UB_i), an initial random centroid $\mathbf{C}_0 = \{C_{10}, C_{20}, \dots, C_{n0}\}$, tolerance $\Delta \mathbf{x}$ and a stopping criterion, such as the maximum number of iterations, the proposed approach consist of:

While the stopping criterion is not satisfied, do:

- {
- Using EA, generate a new centroid $\mathbf{C} = \{C_1, C_2, \dots, C_n\}$
- Using tolerance $\Delta \mathbf{x}$, generate the hyper-box $\mathbf{C} \pm \Delta \mathbf{x}$.
- Evaluate all constraint functions on the hyper-box, as interval functions.
- If the generated hyper-box does not belong to the feasible region then
 - Evaluate the yield as the (estimated) hyper-volume of $F \cap B$ and
 - Fitness (\mathbf{C}) = hyper-volume of $F \cap B$
- Else Fitness (\mathbf{C}) = 100 % and STOP
- }

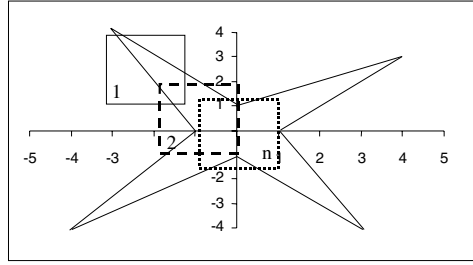


Fig. 3. The proposed approach: Box location is evolved by EA from position 1 to position n

Note that two situations can arise if the hyper-box is not included into the feasible region F : 1) The interval evaluation has produced an overestimation but the hyper-box is really contained in the feasible region, or 2) The generated hyper-box is really not contained in the feasible region.

In the first case the evaluation of the hyper-volume will produce a yield of 100 %. In the second case, the evaluation will produce the estimated yield for C .

Figure 3 shows the proposed approach for the case of two variables. The region inside the star is the feasible region F . Given $\Delta \mathbf{x} = [1, 1.25]^t$, box 1 is randomly located. Note that part of the area of this box is outside F . Next, using EA, box 2 (dashed line) is located. Again part of the box area is outside the feasible region. Finally the location of the box is evolved until box n (dotted line) is reached: its area is almost all inside the feasible region.

4.2 Hyper-Volume Evaluation

To calculate the hyper-volume three approaches can be used. The first one is based on a deterministic exploration [14]: A regular grid is laid over the tolerance region and the evaluation of constraints is performed at each point. The second approach is based on a statistical exploration based on a Monte Carlo procedure [14]: a point in B is generated and it is evaluated if it also belongs to F . The sample can be selected using a random or an importance procedure [15]. The process is repeated N times.

The third approach is based on guaranteed numerical methods for approximating sets, such as the one suggested by Jaulin et al [16]. In this approach the region $F \cap B$ is approximated by covering it with simple non-overlapping subsets of \mathcal{R}^n , such as hyper-boxes. The procedure suggested in [16], called SIVIA (Set Inverter Via Interval Analysis), is able to determine two sets X_{\min} and X_{\max} such that $X_{\min} \subset F \cap B \subset X_{\max}$.

The two covering subsets X_{\min} and X_{\max} are obtained by starting with an initial large hyperbox X_o , which is then iteratively reduced until the pair X_{\min} and X_{\max} defines a neighbourhood of $F \cap B$ with a diameter that can be chosen arbitrarily small.

Even if SIVIA is a guaranteed method, a trade-off could be required since interval procedures are more time consuming than deterministic or statistical procedures.

Figure 4 shows how the three procedures are used to evaluate the area of the region inside the star. For case a) to c), the area is estimated as the percentage of total points inside the region. For case d), the area is estimated as the sum of the area of each of the rectangles.

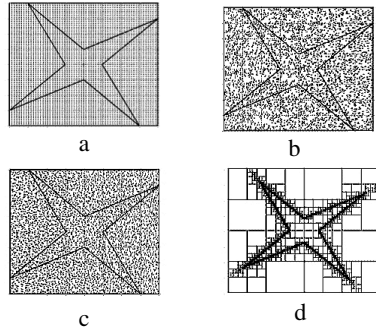


Fig. 4. Area Approximation: a) Deterministic Approach, b) Random sampling, c) Importance sampling, d) SIVIA

5 Computational Example

The temperature controller circuit [2] shown in Figure 1 is analysed. Assume that the design requires that R_{T-on} belongs to the interval [2.9, 3.1] k Ω . Table 1 shows the final interval result for the MIB approach obtained in [7]. These variable ranges produce an output R_{T-on} belonging to: [2.90573, 3.09999] k Ω . However note that the average tolerance of each physical element for this robust design is less than 1 %, which means that the design could be expensive. For this reason, we investigate possible alternative designs with different predefined component tolerance.

In the first yield evaluation, the starting tolerances shown in Table 1 are selected, with 3000 samples for the hyper-volume evaluation. The average yield is only 7.95 %, which indicates that the use of such tolerances is not recommended. Although the evaluations of the hyper-volume are almost equal, the computational time for the SIVIA procedure is near 4000 % higher than the other approaches.

To analyse how the component tolerance influences the design quality, five centring cases were analysed with tolerances of 1, 2.5, 5, 10 and 25 %, for each component. Figure 5 shows the behaviour of the average yield for the selected tolerances (after 20 trials). Even if both EA produce similar results, the (1+1) strategy shows an

Table 1. MIB Robust design for the temperature controller circuit [7]

| VARIABLE | Starting Interval | Mid point | Starting Tolerance (%) | Final Interval | Final Tolerance (%) |
|---------------------|-------------------|-----------|------------------------|-----------------|---------------------|
| R_1 (k Ω) | [0.5,1.5] | 1.0 | 50.00 | [0.994,1.005] | 0.500 |
| R_2 (k Ω) | [6,12] | 9 | 33.33 | [8.956,9.044] | 0.489 |
| R_3 (k Ω) | [2,6] | 4 | 50.00 | [3.973,4.027] | 0.675 |
| R_4 (k Ω) | [16,48] | 32 | 50.00 | [31.428,32.572] | 1.788 |
| E_1 (Volt) | [7.5,9.5] | 8.5 | 11.76 | [8.427,8.573] | 0.859 |
| E_2 (Volt) | [4.5,7.5] | 6.0 | 25.00 | [5.946,6.0535] | 0.892 |

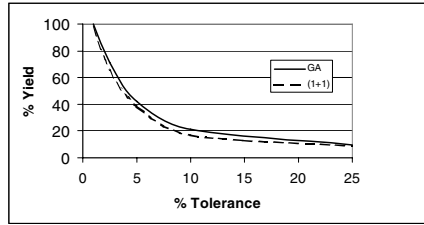


Fig. 5. Average yield as a function of component tolerance

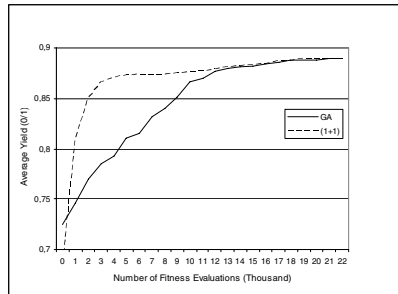


Fig. 6. ES(1+1) and GA convergence for case 1: Component tolerance of 1%

initial faster convergence than GA. For example figure 6 shows the behaviour for case 1 (1 % tolerance). Similar results were observed for the other cases.

From Fig. 5, for example, if the DM is looking for a design with a yield greater than 80 %, then component tolerances must be less than 2.5 %.

6 Conclusions

This paper proposes a promising approach based on the use of Evolutionary Algorithms, Interval Arithmetic and procedures to estimate the yield design as an alternative technique to obtain optimal centring design. The approach is direct and intuitive.

The excellent results obtained suggest that the proposed approach has great potential in dealing with difficult system design problems.

The use of a guaranteed method based on Interval Arithmetic such as SIVIA needs more time than deterministic or statistical approaches. However, even if a deterministic or statistical sampling procedure could not guarantee the results, the differences in evaluating the hyper-volume are small. Both ES (1+1) and GA produces similar results, but ES (1+1) shows an initial faster convergence than GA.

The problem can be easily extended to consider the situation where DM defines a minimum yield, say 95 %. In this case, the problem can be solved looking for \mathbf{C} and tolerances $\Delta\mathbf{x}$, which produces at least, the required yield.

References

1. E.M. Hendrix, C.J. Mecking and Th.H.B. Hendriks: "Finding Robust Solutions for Product Design Problems", *European Journal of Operational Research*, 92, 1996, pp. 28-36
2. S. Hadjihassan, E. Walter, L. Pronzato, Quality Improvement via Optimisation of Tolerance Intervals During the Design Stage, in *Applications of Interval Computations*, Ed. Kearfott R.B., Kreinovich V., Kluwer Publishers, Dordrecht, 1996
3. M. Li et al., A multi-objective genetic algorithm for robust design optimization, *GECCO Proceedings*, pp.771-778, 2005
4. Y. Jin, J. Branke, Evolutionary Optimization in Uncertain Environments - A Survey, *IEEE Transactions on Evo. Comp.* vol.9, no. 3, June 2005
5. Tsutsui S., Ghosh A., "Genetic Algorithms with a Robust Solution Searching Scheme", *IEEE Transaction on Evolutionary Computation*, Vol. 1, No. 3, pp. 201-208, 1997
6. D.H. Loughlin, S. Ranjithan, The Neighborhood constraint method: A Genetic Algorithm-Based Multiobjective Optimization Technique. In Thomas Bäck, editor, *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 666-673, San Mateo, California, July 1997, Michigan State University, Morgan Kaufmann Publishers.
7. C. Rocco, A Hybrid Approach based on Cellular Evolutionary Strategies and Interval Arithmetic to perform Robust Designs", 2nd European Workshop on Evolutionary Algorithms in Stochastic and Dynamic Environments, Lausanne, Switzerland, Lecture Notes in Computer Science LNCS 3449, pp. 623-628, Springer-Verlag
8. R. Moore: *Methods and Applications of Interval Analysis*, SIAM Studies in Applied Mathematics, Philadelphia, 1979
9. A. Neumaier: *Interval Methods for Systems of Equations*, Cambridge Univ. Press, 1990
10. E. Hansen: "Global Optimization Using Interval Analysis", Marcel Dekker Inc., New York, 1992
11. H.P Schwefel, Th. Back: "Evolution Strategies I: Variants and their computational implementation", in J. Periaux and G. Winter (Eds.), *Genetic Algorithm in Engineering and Computer Science*, John Wiley & Sons, 1995.
12. F. Kursawe: "Towards Self-Adapting Evolution Strategies", Proc. Of the Tenth International Conference on Multiple Criteria Decision Making, G Tzeng and P. Yu (Eds.), Taipei 1992
13. Z. Michalewicz, *Genetic Algorithms + Structures Datas = Evolution Programs*, Second Edition, Springer-Verlag, 1994.
14. R. Spence, R. Singh S.: *Tolerance Design of Electronic Circuits*, Addison-Wesley Pub. Co., Wokingham, England, 1988
15. Saltelli, K. Chang, M. Scott.: *Sensitivity Analysis*, John Wiley & Sons, Chichester, 2000
16. L. Jaulin, M. Kieffer, O. Didrit, E. Walter: *Applied Interval Analysis*, Springer, London 2001

A New Self-adaptative Crossover Operator for Real-Coded Evolutionary Algorithms

Manuel E. Gegúndez¹, Pablo Palacios², and José L. Álvarez²

¹ Department of Mathematics
University of Huelva, Huelva, Spain
² Department of Computer Science
University of Huelva, Huelva, Spain
alvarez@uhu.es

Abstract. In this paper we propose a new self-adaptative crossover operator for real coded evolutionary algorithms. This operator has the capacity to simulate other real-coded crossover operators dynamically and, therefore, it has the capacity to achieve exploration and exploitation dynamically during the evolutionary process according to the best individuals. In other words, the proposed crossover operator may handle the generational diversity of the population in such a way that it may either generate additional population diversity from the current one, allowing exploration to take effect, or use the diversity previously generated to exploit the better solutions.

In order to test the performance of this crossover, we have used a set of test functions and have made a comparative study of the proposed crossover against other classic crossover operators. The analysis of the results allows us to affirm that the proposed operator has a very suitable behavior; although, it should be noted that it offers a better behavior applied to complex search spaces than simple ones.

1 Introduction

Evolutionary Algorithms are general purpose search algorithms which are inspired by natural evolution to evolve solutions to complex problems [1][2]. These algorithms maintain a population of individuals (coded candidate solutions to the concrete problem) which are manipulated by three operators: selection, crossover and mutation. In brief, the selection process drives the searching towards the regions of the best individuals, the crossover operator combines individuals to generate better offspring and the mutation operator randomly alters part of one individual increasing the structural diversity of the population.

Although in the origins of the Evolutionary Algorithms, the individuals were coded using the binary alphabet, over the past few years, however, there have been a surge of studies related to real-parameter genetic algorithms due to an increasing interest in solving real-world optimization problems, for example chemometric problems [3], neural networks [4], aerospace design [5], biotechnology [6],

control [7], economic [8], signal processing [9], microware [10], industrial electronics [11], industrial engineering [12], tomography [13], water resources management [14] and constrained parameter optimization problems [15].

This new representation requires adaptation changes in the traditional genetic operators. In this sense, the crossover operator has required bigger adaptation due to the necessity of procedures for the recombination of real values. However, in our opinion, the current real-coded crossover operators only work appropriately for some domains, since they are not adapted to the domain of the problem.

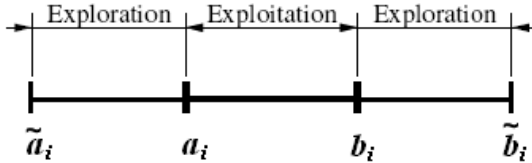


Fig. 1. Geometric representation of the effect of the crossover operator

Generally, the crossover operator combines two individuals (parents) to generate two new individuals (offspring). Thus, let be two individuals $a = (a_1, a_2, \dots, a_n)$ and $b = (b_1, b_2, \dots, b_n)$, the offspring $y = (y_1, y_2, \dots, y_n)$ are generated such that $y_i = f(a_i, b_i)$. Geometrically, the effect of the crossover operator is shown in the Fig. 1, where the interval $[a_i, b_i]$ is the exploitation area and the intervals $[\tilde{a}_i, a_i]$ and $[b_i, \tilde{b}_i]$ are the exploration areas.

In this work, we present a new self-adaptive real-coded crossover operator which has the capacity of adaptation to the domain of the problem to solve. In summary, this operator can establish the areas of exploitation and exploration in an appropriate way to the problem for a better adaptation of the individuals.

The rest of the paper is organized as follows. In Sect. 2, we present the more relevant related works. In Sect. 3, we enunciate the principal features of the new crossover operator. In Sect. 4, we describe a genetic algorithm which use the proposed crossover operator. In Sect. 5, we compare the performance of new operator against other crossover operators by means of several fitness functions. Finally, Sect. 6 states the conclusions and the future research lines related to this work.

2 Related Works

Among numerous studies on development of different recombination operators, Uniform crossover (UX), Blend crossover (BLX) and Simulated Binary crossover (SBX) are commonly used. A number of other recombination operators, such as Arithmetic crossover, Geometric crossover, extended crossover are similar to BLX operator. A detailed study of many such operators can be found elsewhere [16].

In this section, we define principal real-coded crossover operator that appear in the literature about Evolutionary Algorithm. Thus, let A and B be the two individuals selected to crossover, with $A = (a_1, a_2, \dots, a_n)$ and $B = (b_1, b_2, \dots, b_n)$, the offsprings are generated according to the next crossover operator:

Uniform crossover (UX). [19] Two offspring X and Y are generated, so that the genes of the parents (a_i and b_i) are transferred randomly for each offspring. Equation 1 shows this process.

$$x_i = \begin{cases} a_i & \text{if } c = 1 \\ b_i & \text{if } c = 0 \end{cases} \quad y_i = \begin{cases} b_i & \text{if } c = 1 \\ a_i & \text{if } c = 0 \end{cases} \quad (1)$$

Arithmetical crossover (AX). [15] Two offspring X and Y are generated, whose genes x_i and y_i are obtained from the genes of the parents (a_i and b_i) by equation 2, where λ is a constant.

$$\begin{aligned} x_i &= \lambda \cdot a_i + (1 - \lambda) \cdot b_i \\ y_i &= \lambda \cdot b_i + (1 - \lambda) \cdot a_i \end{aligned} \quad (2)$$

item[Geometric crossover (GX).] [19] Two offspring X and Y are generated, whose genes x_i and y_i are obtained from the genes of the parents (a_i and b_i) by equation 3, where ω is defined in the interval $[0, 1]$.

$$\begin{aligned} x_i &= a_i^\omega \cdot b_i^{(1-\omega)} \\ y_i &= b_i^\omega \cdot a_i^{(1-\omega)} \end{aligned} \quad (3)$$

Blend crossover (BLX- α). [20] One offspring X is generated, whose genes x_i are obtained randomly from the genes of the parents (a_i and b_i) from the interval $[Min - I \cdot \alpha, Max + I \cdot \alpha]$, where $Min = \min(a_i, b_i)$, $Max = \max(a_i, b_i)$, $I = Max - Min$ and α is defined in the interval $[0, 1]$.

Simulated Binary crossover (SBX- α). [21] Two offspring X and Y are generated, whose genes x_i and y_i are obtained by equation 4, where α is a value within of the interval $[0, \infty)$.

$$\begin{aligned} x_i &= \frac{1}{2} \cdot [(1 - \alpha) \cdot a_i + (1 + \alpha) \cdot b_i] \\ y_i &= \frac{1}{2} \cdot [(1 + \alpha) \cdot a_i + (1 - \alpha) \cdot b_i] \end{aligned} \quad (4)$$

3 GPAX Crossover Operator

In this section, we deal with the main aspects of the new crossover operator, called *GPAX*, describing the general mechanism and their principal features.

Let us assume that $A = (a_1, a_2, \dots, a_n)$ and $B = (b_1, b_2, \dots, b_n)$ are two individuals that have been selected to apply the crossover operator to them and that $i \in \{1, \dots, n\}$ is the position of the real gene x_i in the offspring $X = (x_1, x_2, \dots, x_n)$ to be determined by the *GPAX* crossover operator.

Thus, the *GPAX* parabolic adaptative crossover function is based on parabolic probability density function which is defined starting from two parent solutions

a_i and b_i , with $a_i < b_i$, in the interval $[a_i - \frac{b_i - a_i}{2}, b_i + \frac{b_i - a_i}{2}]$ (denoted $[\tilde{a}_i, \tilde{b}_i]$). This density function depends on two real parameters α and β , both defined in the interval $(0, 1)$. Equation 5 shows the *GPAX* parabolic adaptative crossover function. The probability density function $f_{a_i, b_i}(x, \alpha, \beta)$ generates a cumulative distribution function $\mathcal{F}_{a_i, b_i}(x, \alpha, \beta)$ which also depends of α and β parameters, as is shown in equation 6. The cumulative distribution function is used to determine randomly the genes of the offspring by means of the function $\mathcal{F}_{a_i, b_i}^{-1}(y, \alpha, \beta)$.

$$f_{a_i, b_i}(x, \alpha, \beta) : [\tilde{a}_i, \tilde{b}_i] \times (0, 1) \times (0, 1) \rightarrow \mathcal{R} \quad (5)$$

$$\mathcal{F}_{a_i, b_i}(x, \alpha, \beta) : [\tilde{a}_i, \tilde{b}_i] \times (0, 1) \times (0, 1) \rightarrow [0, 1] \quad (6)$$

$$x_i = GPAX_{\alpha, \beta}(a_i, b_i) = \mathcal{F}_{a_i, b_i}^{-1}(\nu, \alpha, \beta) \quad (7)$$

The procedure of creating an offspring X from two parent A and B is described as follows. Given the α and β parameters, for each children solution, a random number ν is uniformly chosen in the interval $[0, 1]$. Thereafter, from the inverse of the specified probability distribution function $\mathcal{F}_{a_i, b_i}(x, \alpha, \beta)$, the gen x_i is found so that the area under the probability density function from \tilde{a}_i to x_i is equal to the chosen random number ν (i.e. $\nu = \mathcal{F}_{a_i, b_i}(x_i, \alpha, \beta)$). So x_i is calculated by means of the equation 7, where ν is a value random which is uniformly selected in the interval $[0, 1]$.

Table 1. Some equivalences between *GPAX* and other crossover operators according to the values of alpha and beta parameters

| α | β | Crossover Operators |
|---------------|----------------|---------------------|
| ≈ 0.0 | ≈ 0.67 | $\approx BLX$ |
| ≈ 0.9 | ≈ 0.40 | $\approx SBX$ |
| ≈ 1.0 | ≈ 1.00 | $\approx UX$ |

Thus, *GPAX* crossover operator is able to produce exploration or exploitation (at different degrees) depending on the values of the parameters α (the exploration parameter) and β (the exploitation parameter). The exploration parameter α determines the external zones in the $[\tilde{a}_i, \tilde{b}_i]$ interval, subintervals $[\tilde{a}_i, a_i]$ and $[b_i, \tilde{b}_i]$, modeling the probability density function in both subintervals. The exploitation parameter β determines the intermediate zone central in the $[\tilde{a}_i, \tilde{b}_i]$, subinterval $[a_i, b_i]$, the probability density function in this subinterval.

Initially, as suggest blend crossover *BLX*- α (see [20]), *GPAX* randomly picks a solution in the range $[\tilde{a}_i, \tilde{b}_i]$. Generally, it should be noted that *GPAX* procedure has the capacity to simulate the real-coded crossover operators according to the α and β parameters. Table 1 shows some equivalences between *GPAX* and others crossover operators. Fig. 2 and 3 show some examples of the probability density functions and the cumulative distribution functions, $f_{-1,1}(x, \alpha, \beta)$ and

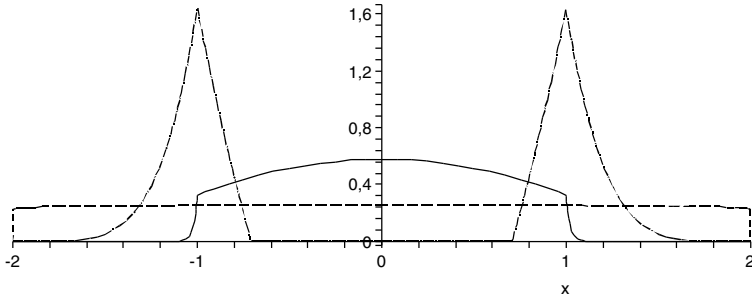


Fig. 2. Examples of the Probability Density Functions according to the values of alpha and beta parameters for $\mathcal{F}(x, 0.9, 0.4)$, $\mathcal{F}(x, 0.01, 0.67)$ and $\mathcal{F}(x, 0.99, 0.9)$ (dash-dot, dash and solid, respectively)

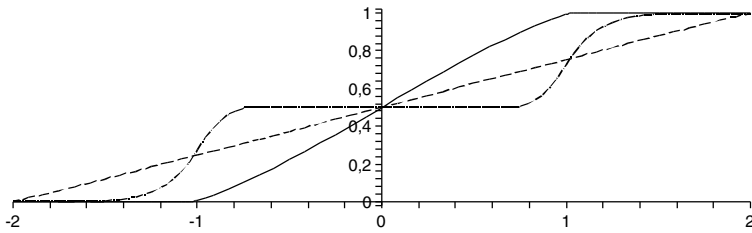


Fig. 3. Examples of the Cumulative Distribution Functions according to the values of alpha and beta parameters for $\mathcal{F}(x, 0.9, 0.4)$, $\mathcal{F}(x, 0.01, 0.67)$ and $\mathcal{F}(x, 0.99, 0.9)$ (dash-dot, dash and solid, respectively)

$\mathcal{F}_{-1,1}(x, \alpha, \beta)$, where α and β have different values. These figures show, graphically, as *GPAX* simulate other crossover operator according to the α and β parameters.

Although *GPAX* has the capacity to simulate some real-coded crossover operators; however, it should be noted that the principal feature of *GPAX* is the possibility to exchange from a model to another by changing the exploitation and exploration parameters during the evolutionary process. In other words, *GPAX* may handle the generational diversity of the population in such a way that it may either generate additional diversity population from the current one, therefore exploration takes effect, or use the diversity previously generated to exploit the better solutions.

This self-adaptive feature of *GPAX* allows to develop a new dynamic model of crossover which adapts to the current problem according to the evolution and the population in each generation.

4 Evolutionary Algorithm

Besides the recombination operator, researchers have also realized the importance of a different genetic algorithm model. In this section, we explain the

Evolutionary Algorithm

1. $t \leftarrow 0$
 2. Initialize Population P_t
 3. Initialize α and β parameters
 4. Evaluate Initial Population P_t
 5. While (numGenerations < MAXGENERATIONS)
 6. $t \leftarrow t + 1$
 7. Select P_t from P_{t-1}
 8. Crossover P_t using *GPAX*
 9. Mutate P_t
 10. Evaluate P_t
 11. $\alpha = \alpha'$ and $\beta = \beta'$, where α' and β' are the parameters of the best individual
 12. End
- End
-

Fig. 4. Evolutionary Algorithm

properties of the developed Evolutionary Algorithm in order to study the behavior of the *GPAX* crossover operator. In our study we have taken a traditional Evolutionary Algorithm model which, obviously, incorporates a mechanism of adaptation of the parameters α and β of the new crossover operator from one generation to the next.

Concretely, besides the traditional parameters, two new parameters (α and β) are used. Next, the crossover procedure is detailed:

1. From the current population, select two parents $A = (a_1, a_2, \dots, a_n)$ and $B = (b_1, b_2, \dots, b_n)$.
2. Generate two offspring $X = (x_1, x_2, \dots, x_n)$ and $Y = (y_1, y_2, \dots, y_n)$ from the chosen 2 parents using a *GPAX* crossover
 - (a) Calculate $x_i = GPAX_{\alpha, \beta}(a_i, b_i) = \mathcal{F}_{a_i, b_i}^{-1}(\nu, \alpha, \beta)$, where ν is a random value which is uniformly selected in the interval $[0, 1]$
 - (b) Produce $\alpha' = \alpha \pm 0.1$ and $\beta' = \beta \pm 0.1$
 - (c) Calculate $y_i = GPAX_{\alpha', \beta'}(a_i, b_i) = \mathcal{F}_{a_i, b_i}^{-1}(\nu', \alpha', \beta')$, where ν' is a random value which is uniformly selected in the interval $[0, 1]$

Besides, once the entire population have been evaluated the α and β parameters of the best individual will be the new α and β parameters for the next generation.

In addition to this new procedure, the evolutionary process applies an elitist strategy, the roulette-wheel selection operator and the non-uniform mutation operator. Fig. 4 offers the outline of the traditional Evolutionary Algorithm.

5 Experiments

The objective of this section is to make a comparison between *GPAX* crossover and other crossover operators. We have chosen some classic crossovers operators, like Uniform crossover (UX), BLX- α and SBX- α with different α values.

$$f_{Sph}(\vec{x}) = \sum_{i=1}^n x_i^2.$$

$$f_{Ros}(\vec{x}) = \sum_{i=1}^{n-1} (x_{i+1} - x_i^2)^2 + (x_n - 1)^2.$$

$$f_{Sch}(x) = \sum_i^n \left(\sum_{j=1}^i x_j \right)^2.$$

$$f_{Ras}(x) = a * n + \sum_{i=1}^n x_i^2 - a * \cos(\varpi * x_i) \text{ where } a = 10 \text{ and } \varpi = 2\pi.$$

$$f_{Gri}(x) = \frac{1}{d} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

Fig. 5. Test Fitness Functions

This comparative study shows the performance the proposed crossover against other operators using a set of functions. We have considered five frequently used test functions [22]: Sphere model (f_{Sph}) [23], Generalized Rosenbrock's function (f_{Ros}) [23], Schwefel's Problem (f_{Sch}) [24], Generalized Rastrigin's function (f_{Ras}) [25], Griewangk's function (f_{Gri}) ([26], where the dimension of the search space (n) is 25. Fig. 5 shows their formulation. These functions are: f_{Sph} , a continuous, strictly convex, and unimodal function; f_{Ros} , a continuous and unimodal function, with the optimum located in a steep parabolic valley with a flat bottom. f_{Sch} , a continuous and unimodal function. Its difficulty concerns the fact that searching along the coordinate axes only gives a poor rate of convergence, since the gradient of f_{Sch} is not oriented along the axes; f_{Ras} , a scalable, continuous, and multimodal function, which is made from f_{Sph} by modulating it with $a * \cos(\omega * x_i)$ and f_{Gri} , a continuous and multimodal function. This function is difficult to optimize because it is non-separable and the search algorithm has to climb a hill to reach the next valley.

We carried out our experiments using the following parameters: the population size is 60 individuals, the crossover probability 0.6, the probability of mutation 0.125, elitist selection and the sampling model was roulette-wheel selection operator. We executed all the algorithms 30 times, each one with 100.000 evaluations.

The results are presented in Tables 2 and 3. For each function, the average values and the best one are shown. Next, we comment about the results of GPAX for each functions: for f_{Sph} function, GPAX does not show a good behavior, the average and the best solution are worse than for the other operators.; for f_{Sch} function, it obtains the best average and the best solution; for f_{Ras} function, it obtains the sixth average and the best solution; for f_{Gri} function, it obtains the best average and the third best solution; and for f_{Ros} function, it show a good behavior, it obtains the best average and the best solution.

Table 2. Results for Sphere model (f_{Sph}) and Schwefel's Problem (f_{Sch})

| Crossover | f_{Sph} | | f_{Sch} | |
|-----------|-----------|----------|-----------|----------|
| | Average | Best | Average | Best |
| UX | 1.06E-08 | 1.73E-09 | 7.21E+02 | 2.64E+02 |
| BLX-0 | 1.28E-08 | 2.54E-09 | 4.00E+01 | 9.86E+00 |
| BLX-0.3 | 7.51E-11 | 1.27E-11 | 3.37E+01 | 8.04E+00 |
| BLX-0.5 | 6.31E-06 | 6.12E-07 | 1.36E+03 | 5.76E+02 |
| SBX-2 | 1.97E-09 | 4.38E-10 | 7.56E+00 | 7.09E-01 |
| SBX-5 | 2.76E-10 | 6.00E-11 | 9.54E+01 | 1.14E+01 |
| GPAX | 4.00E-02 | 3.92E-06 | 7.98E-02 | 5.90E-10 |

Table 3. Results for Generalized Rastrigin's function (f_{Ras}), Griewangk's function (f_{Gri}) and Generalized Rosenbrock's function (f_{Ros})

| Crossover | f_{Ras} | | f_{Gri} | | f_{Ros} | |
|-----------|-----------|----------|-----------|----------|-----------|----------|
| | Average | Best | Average | Best | Average | Best |
| UX | 6.96E-01 | 9.94E-01 | 2.22E+02 | 2.15E-06 | 5.10E+01 | 1.60E+00 |
| BLX-0 | 4.47E+00 | 9.94E-01 | 1.55E-02 | 2.26E-06 | 2.22E+01 | 2.05E+01 |
| BLX-0.3 | 7.86E+00 | 4.97E+00 | 1.54E-02 | 1.10E-08 | 2.18E+01 | 1.92E+01 |
| BLX-0.5 | 8.72E+01 | 6.08E+01 | 5.29E-01 | 5.06E-03 | 2.61E+01 | 2.09E+01 |
| SBX-2 | 6.96E+00 | 1.36E+01 | 1.91E-02 | 3.22E-07 | 2.99E+01 | 1.74E+01 |
| SBX-5 | 7.13E+00 | 2.98E+00 | 2.32E+02 | 8.69E-08 | 3.90E+01 | 1.64E+00 |
| GPAX | 2.31E+01 | 2.94E-03 | 3.53E-03 | 8.49E-07 | 1.60E+01 | 4.00E-02 |

6 Conclusions

This paper presented a real-coded adaptative crossover operator, called *GPAX* (Genetic Parabolic Adaptative crossover operator). The *GPAX* crossover operator is able to produce exploration or exploitation (at different degrees) depending on the values of the parameters α (the exploration parameter) and β (the exploitation parameter). Although *GPAX* has the capacity to simulate some real-coded crossover operators; however, it should be noted that the principal feature of *GPAX* is the possibility to switch from one model to another by changing the exploitation and exploration parameters during the evolutionary process.

Looking back over the results, we may observe that *GPAX* shows a good behavior, except with f_{Sph} function. As it is expected, *GPAX* does not show an efficient progress on simple fitness functions; however, it works better on complex search spaces, where it achieves the best average and/or the best solution. Concretely, the best results are obtained for f_{Sch} and f_{Ros} . These promising results motivate the deep study about this new adaptative crossover operator, focusing in the α and β parameters and their tuning in the Evolutionary Algorithm.

References

1. Holland, J.H.: *Adaptation in Natural and Artificial Systems*. University of Michigan Press (1975)
2. Goldberg, D.: *Genetic Algorithm in Search, Optimisation, and Machine Learning*. Addison Wesley Publishing Company. Ann Arbor-Michigan USA (1989).
3. Lucasius C. B., Kateman G.: Applications of genetic algorithms in chemometrics. Proc of the 3rd Int Conf on Genetic Algorithms. Morgan Kaufmann Publishers (1989) 170-176
4. Zamparelli M.: Genetically trained cellular neural networks. *Neural Networks* **10**(6) (1997) 1143-1151
5. Oyama A, Obayashi S, Nakamura T.: Real-coded adaptive range genetic algorithm applied to transonic wing optimization. *Appl Soft Computing* **1**(3) (2001) 179-187
6. Roubos J.A., van Straten, van Boxtel: An evolutionary strategy for fed-batch bioreactor optimization; concepts and performance. *Journals Biotechnol*, vol. 67(2-3) (1999) 173-187
7. Kawabe T, Tagami T. A real coded genetic algorithm for matrix inequality design approach of robust PID controller with two degrees of freedom. In Proc of the 1997 IEEE Int Symp on Intelligent Control. (1997) 119-124
8. Duffy, J., McNelis, P. D.: Approximating and simulating the stochastic growth model: Parameterized expectations, neural networks, and the genetic algorithm. *Journal Econ Dyn Control* **25**(9) (2001) 1273-1303
9. Harris S.P., Ifeachor E.C. Automatic design of frequency sampling filters by hybrid genetic algorithm techniques. *IEEE Trans Sig Process* **46**(12) (1998) 3304-3314
10. Caorsi S., Massa A., Pastorino M.: A computational technique based on a real-coded genetic algorithm for microwave imaging purposes. *IEEE Trans Geosci Remote Sens* **38**(4) (2000) 1697-1708
11. Shi K.L., Chan T.F., Wong Y.K., Ho S.L.: Speed estimation of an induction motor drive using an optimized extended Kalman filter. *IEEE Trans Indust Electron* **49**(1) (2002) 124-133
12. Azariadis P.N., Nearchou A.C., Aspragathos N.A.: An evolutionary algorithm for generating planar developments of arbitrarily curved surfaces. *Comp Industry* **47**(3) (2002) 357-368
13. Turcanu C.O., Craciunescu, T.: A genetic approach to limited data tomographic reconstruction of time-resolved energy spectrum of short-pulsed neutron sources. *Pattern Recognition Letter* **23**(8) (2002) 967-976
14. Chang, F.J.: Real-coded genetic algorithm for rule-based flood control reservoir management. *Water Resource Management* **12**(3) (1998) 185-198
15. Michalewicz, Z.: *Genetics Algorithms + Data Structures = Evolution Programs*, WNT, Warsaw (1996)
16. Herrera, F., Lozano, M., Verdegay, J. L.: Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. *Artificial Intelligence Review* **12**(4) (1998) 265-319
17. Deb, K.: *Multi-objective optimization using evolutionary algorithms*. Wiley. (2001)
18. Wright, A. H.: *Genetic Algorithms for Real Parameter Optimization*. Foundations of genetic algorithms. (1991)
19. Syswerda, G.: Uniform crossover in genetic algorithms, in ICGA-89, J. D. Schaffer, Ed. Morgan Kaufmann. (1989)
20. Eshelman, L. J., Caruana, R. A., Schaffer, J. D.: Biases in the Crossover Landscape (1997)

21. Agrawal, R. B.: Simulated binary crossover for real-coded genetic algorithms. Indian Institute of Technology, Kanpur (1995)
22. Lozano, M. et al.: Real Coded Memetic Algorithms with Crossover Hill-Climbing. *Evolutionary Computation* **12**(3) (2004) 273–302
23. De Jong, K.E.: An analysis of the behavior of a class of genetic adaptive systems. University of Michigan Press, Ann Arbor. (1975)
24. Schwefel, H.P.: Numerical Optimization of Computer Models. Wiley. (1981)
25. Torn, A., Zilinskas, A.: Global Optimization. Springer-Verlag, Berlin. (1989)
26. Griewangk, A.O.: Generalized Descent for Global Optimization (34) (1981)
27. Beyer, H.-G., Deb, K.: On self-adaptive features in real-parameter evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, **5**(3) (2001) 250–270
28. Kita, H.: A comparison study of self-adaptation in evolution strategies and real-coded genetic algorithms. *Evolutionary Computing Journal* **9**(2) (2001) 223–241

Wavelet Enhanced Analytical and Evolutionary Approaches to Time Series Forecasting

Bartosz Kozłowski

Institute of Control and Computation Engineering,
Warsaw University of Technology,
Nowowiejska Str. 15/19, 00-665 Warsaw, Poland
b.kozlowski@elka.pw.edu.pl

Abstract. This paper provides two methodologies for forecasting time series. One of them is based on the Wavelet Analysis and the other one on the Genetic Programming. Two examples from finance domain are used to illustrate how given methodologies perform in real-life applications. Additionally application to specific classes of time series, seasonal, is discussed.

1 Introduction

For many years numerous attempts have been made to solve a number of complex problems existing in finance domain. Different computational techniques have been applied, including Artificial Intelligence (AI) methods and nature inspired computing (e.g. [2], [3]). This paper will focus on one of these problems, namely forecasting of Time Series (TS). One of the first attempts to forecasting was made in Japan in the 14th century [13]. In the 19th century Charles Dow presented some of his thoughts in a series of publications which constitute a knowledge called the Dow Theory nowadays [16]. This theory provided a background for thoroughly explored Technical Analysis.

Wavelet Analysis (WA), which is the first methodology applied in here, appeared in the beginning of the 20th century in the discussion on orthogonal systems of functions [7]. Alfred Haar introduced a function which modified version is considered today as a first basic wavelet function. WA started to develop rapidly in the nineties of the 20th century. Wavelet-based methods were found to be useful for solving many problems including: analysis and synthesis of TS [14] (e.g. for TS denoising [5], [10]), image processing [15] (e.g. compression) and data mining [11].

Evolutionary Algorithms (EA) [6], [12] have recently become one of the most popular nature inspired approaches to solving complex real-life optimization problems (e.g. [4]). EA is a computational methodology used for search and optimization which is based on principles of Darwinian natural selection and genetics. In this paper Genetic Programming (GP) will be used. GP is one of main streams of EA and was proposed by Koza [8]. GP has been successfully applied in solving a number of problems in various domains.

This paper is organized as follows. The next section gives a basic view on the TS forecasting problem. In section 3 two forecasting algorithms based on using WA and GP are described. It also provides a seasonal TS model applied in the second experimental case. Section 4 presents and compares results of case studies obtained with the use of previously presented forecasting algorithms. The last section concludes this paper.

2 Time Series Forecasting Problem

The idea of forecasting is to assume what will be the state of the phenomenon in the future based on the knowledge of its “behavior” in the past. If one possesses good knowledge of this phenomenon he/she may apply various model-based or rule-based forecasting approaches. However, if one does not, most commonly extrapolation methods are used. Nevertheless, it is often practicable to mix the model-based and the extrapolation methodologies. Also a number of widely applied forecasting approaches are based on statistical observations [1].

In this paper an overview of two methodologies from the extrapolation area and a mixed approach is provided. It is assumed that a phenomenon is described by a set \mathbf{X} of observations X (TS \mathbf{X}). The TS forecasting problem is solved in two steps. In the first step function A is found which approximates the TS \mathbf{X} . In the second step this function is extrapolated to find the subsequent, future values of the TS \mathbf{X} .

3 Methodologies

3.1 Analytical Approach with Wavelet Analysis

About Wavelets. Wavelet Transform (WT) provides a representation of a TS in a different domain. This domain may be considered as a set of series of observation points (wavelet coefficients) derived from the original TS-based on a special single (but scaled and modified) function. Each point in each of these series may be interpreted as a difference of weighted averages in neighboring intervals. A very important (although not necessarily always used) feature of this transform is its inversability, which means that having wavelet coefficients one may derive original data. As WT splits TS into short and long term trends and allows to perform a multiresolutional insight into the TS, it is natural to assume that the results enforced with wavelet analysis may outperform standard analysis approaches. Most of the wavelet-based methodologies practically applied utilize an analysis schema presented in Fig. 1.

After initial analysis of the TS it is transformed with WT. Subsequently an analysis is performed on the resulting wavelet coefficients. Then, optionally, wavelet coefficients are inversely transformed into the original TS space and a closing analysis is performed.

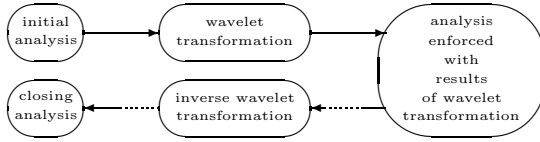


Fig. 1. Schema of the wavelet-based analysis flow

WA-Based Forecasting Algorithm. The essence of wavelet-based forecasting is that approximation and extrapolation of TS are applied repeatedly (for each level) in a wavelet space. Inversely transforming wavelet coefficients together with their forecasts back to the TS space provides a forecast of the original TS. This is organized by Algorithm 3.1.

Algorithm 3.1. WAVELETFORCAST(\mathbf{X}, J)

```

 $\widetilde{\mathbf{W}} \leftarrow$  nonorthonormal DWT of  $\mathbf{X}$ 
for each  $\widetilde{\mathbf{W}}_j \in \widetilde{\mathbf{W}}, j \in J$ 
do
   $\begin{cases} \widetilde{E}_j \leftarrow \text{extrapolation of } \widetilde{\mathbf{W}}_j \\ \widetilde{\mathbf{W}}'_j \leftarrow \widetilde{\mathbf{W}}_j \cup \{\widetilde{E}_j\} \\ \mathbf{W}_j \leftarrow \text{an orthonormal (for level } j) \text{ subset of } \widetilde{\mathbf{W}}'_j \text{ (containing } \widetilde{E}_j) \end{cases}$ 
 $\mathbf{X}' \leftarrow$  Inverse DWT from all  $\mathbf{W}_j$ 
 $E \leftarrow \mathbf{X}' - \mathbf{X}$ 
return ( $E$ )
  
```

In order to forecast the wavelet space coefficients, similar methods to those used in direct TS forecasting have to be applied. Because these methods are applied many times, it is important that one should be aware of the order of their computational complexity.

3.2 Evolutionary Approach with Genetic Programming

About Genetic Programming. GP is an attempt to build a general use system which would build for us computer programs to solve different problems. GP uses evolutionary, genetic mechanisms to suit the given goal. In case of GP an individual, a program (Fig. 2), a single result is represented as a tree.

Nodes are selected from a set of available functions and leaves are filled from a set of given terminals. An initial population in GP consists of a given number of randomly generated trees. Two individuals are crossed by exchanging subtrees in randomly selected nodes. Mutation is accomplished by replacing a randomly selected subtree with a new one generated especially for this purpose. For selection standard Genetic Algorithms' procedures may be applied.

GP-Based Forecasting Algorithm. Forecasting with GP gets down to finding the formula for approximation function A of the TS \mathbf{X} . After finding the

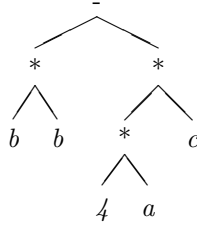


Fig. 2. Parse tree of a program which calculates a Δ of a quadratic equation

approximation function, TS is forecasted by extrapolation with a found function. This is enclosed in Algorithm 3.2

Algorithm 3.2. GPFORECAST(\mathbf{X})

$A \leftarrow$ GP approximation function of \mathbf{X}
 $E \leftarrow$ a value of A in the new time moment u following \mathbf{X}
return (E)

A set of GP terminals consists of original TS observation points. Functions' set may include a variety of functions. Fitness function is formulated as $\sum_u |X(u) - A(u)|$, where $X(u)$ is a value of the original TS in u and $A(u)$ is a value of approximated TS in u . In the evolutionary process GP needs to find the approximation function $A(u)$ which minimizes the value of the error. This function is used in the next step of algorithm to extrapolate forecasted value E .

3.3 Seasonal Time Series Model

Let us consider a TS \mathbf{X} and suppose that it is split into several M disjunctive sub TS \mathbf{X}^m ($m = 1 \dots M$) of an equal length L in a specific manner. Let also TS \mathbf{X} be equal to a sum of two TS \mathbf{F} and \mathbf{G} , where \mathbf{G} is responsible for long term changes in \mathbf{X} and \mathbf{F} corresponds to short term changes. Usually changes in \mathbf{F} are much faster compared to \mathbf{G} , therefore more difficult for proper forecasting. A special case of \mathbf{F} is a situation where \mathbf{F} is a seasonal TS.

Definition 1. *If for each two sub TS \mathbf{F}^{m_1} and \mathbf{F}^{m_2} every pair of points $F_l^{m_1}$ $F_l^{m_2}$, $l = 1, \dots, L$ differs less than assumed ε , then with accuracy of ε TS \mathbf{F} is seasonal and has seasons \mathbf{F}^m .*

Suppose that a subset \mathbf{K} of last K time moments of \mathbf{X} is not known. After splitting \mathbf{X} into seasons \mathbf{X}^m one knows all sub TS \mathbf{X}^m , $m = 1 \dots M - 1$ and TS \mathbf{X}^M without last K observations. A forecast of \mathbf{X}^M on K last time moments is equivalent to a forecast of \mathbf{X} .

By defining \mathbf{H} in locations $l = 1, \dots, L - K$ as $\mathbf{H}^m = \mathbf{X}^m - \mathbf{X}^M$ one gets $\mathbf{G}^m - \mathbf{G}^M$, therefore a slowly changing process easier for forecasting. By using WT one may forecast \mathbf{H}^m on $k \in \mathbf{K}$ points. Now, m -th forecast (\mathbf{X}^{M_m}) of \mathbf{X}^M may be found by $\mathbf{X}^{M_m} = \mathbf{X}^m - \mathbf{H}^m$. Also the final forecast may be calculated

by $X_k^m = \frac{1}{\sum_{m=1}^{M-1} w_m} \sum_{m=1}^{M-1} w_m X_k^{M_m}, k \in \mathbf{K}$, where w_m is a weight of m -th season, equal to e.g. \sqrt{m} .

A more detailed description of forecasting seasonal TS with wavelets may be found in [9]. It is also possible to identify seasonality and its length of a season with wavelet methodology.

4 Sample Experiments – Case Studies

4.1 Overview

In this section an experiment performed on two case studies (Fig. 3) is written up. In both cases TS is forecasted using two methodologies, one of them is GP and the second one is wavelet-based. In the second case study wavelet forecasts are enforced with information about the presence of seasonality in the original TS.

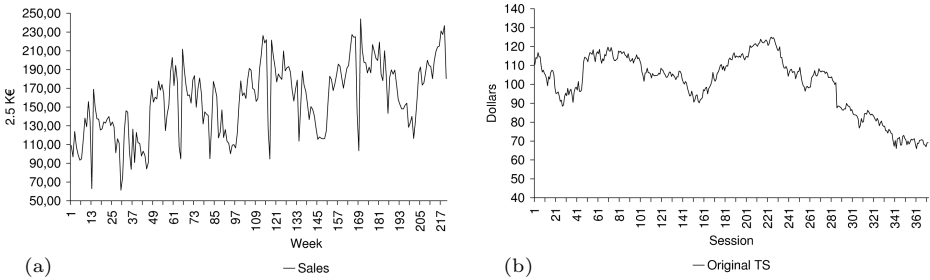


Fig. 3. Original TS – (a) supermarket sales and (b) IBM shares

The set of accepted functions for GP is $\mathbf{F} = \{+, -, *, /, Exp, Sin, Cos, Log, Sqrt, Pow\}$. The set of terminals \mathbf{T} consists of the TS historical data and natural numbers. Fitness function is defined as described in “GP-based forecasting algorithm” section. An initial population of individuals, i.e. parse trees, was created using minimal and maximal values of tree depth equal to 1 and 6, respectively. The purpose of the first set of experiments (not presented in here) was to define initial settings for GP. Finally, GP approximation algorithm was run with following parameters: a size of population - 1000, number of generations - 800, probability of crossover - 0.95, and mutation probability was set to 0.05.

For each experiment a series of one step forward forecasts was performed basing only on historical information. Algorithms were not fed back with results established in previous step. The experiments were performed according to the diagram presented in Fig. 4.

For experiments with market shares a total number of experiments was equal to 110 and in case of sales $e_{max} = 23$. For each experiment (numbered by e) a time window of the same length was selected, meaning that for the next experiment (number $e + 1$) data was extended by one (younger) observation

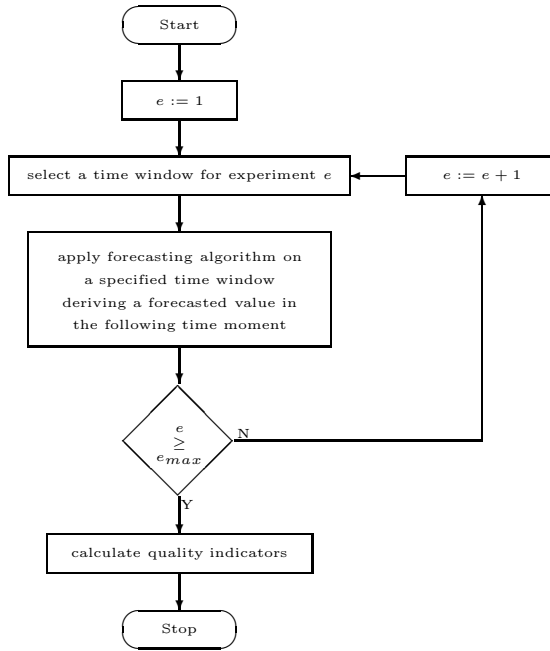


Fig. 4. Diagram describing how experiments were performed

point and truncated by one (the oldest) observation point. Then a forecast was performed and the result was stored in the data store. When there were no more experiments scheduled ($e \geq e_{max}$) the process left the loop. In the end all results were evaluated and analyzed according to four indicators of their quality, namely: standard deviation, maximal error (the biggest of differences between a real-life values and the corresponding forecasted values), minimal error and average error.

4.2 Sales

This case considers the TS of sales in a supermarket chain (Fig. 3a). This TS indicates obvious seasonality, therefore as the second method to describe the wavelet methodology which takes an advantage of this information have been chosen.

The results of forecasts using WA are shown in Fig. 5a and results established with GP in Fig. 5b.

4.3 Market Shares

In this case a TS of prices of IBM's market shares is forecasted. This is a TS with no obvious regularities. GP discovers a solution in the form of a LISP expression which is not presented here due to space limitation. The results of WA forecasts are shown in Fig. 6a and GP forecasts are shown in Fig. 6b.

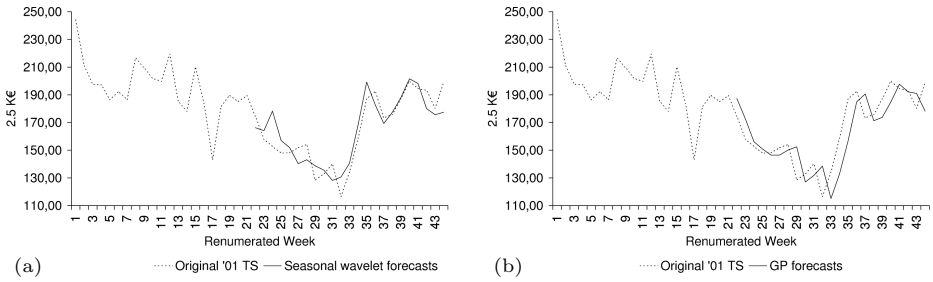


Fig. 5. Results of forecasts of sales using (a) WA and (b) GP

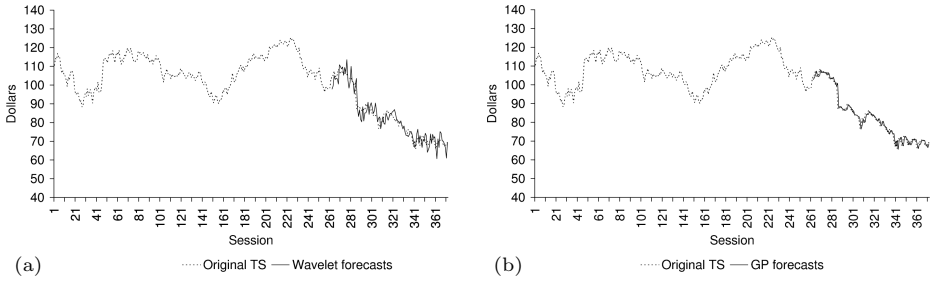


Fig. 6. Results of forecasts of shares using (a) WA and (b) GP

4.4 Results

Both methodologies provide good results (Tab. 1) in general view for both case studies.

In case of sales the GP forecasts provide worse results than wavelet-based approach, what may be observed in Tab. 1b. WA enforced with seasonal approach outperforms GP by over an order of magnitude, as it comes to the average error gauge.

Results of GP forecasts for shares’ prices are better than WA forecasts for all gauges. One can see that only maximal error is relatively close in both methods. In all other cases the results of WA forecast quality tests are at least two times worse than in case of GP forecasts.

Table 1. Quality comparison of (a) market shares forecasts and (b) sales forecasts

| | (a) | | (b) | |
|------------|-------------|-------------|-------------|-------------|
| Gauge | GP | WA | GP | WA |
| std. dev. | 2,742609834 | 5,815448996 | 73758246,89 | 37675615,83 |
| max. error | 0,152457101 | 0,183306337 | 0,180935966 | 0,172322659 |
| min. error | 2,85179E-05 | 0,000310097 | 0,006018841 | 0,004556636 |
| avg. error | 0,018114803 | 0,036521327 | 0,071833853 | 0,056000513 |

5 Conclusions

In this study it has occurred that GP provides better results than the method based on WA when both applied directly. Nevertheless, WA method has an advantage in a form of its very low computational complexity. While GP requires time and relatively big amount of computational resources, wavelet-based methodologies' calculations are performed almost on-the-fly. In case when WA-based method, enforced with additional information about seasonality, is applied it provides better results than GP. Therefore, one may conclude that GP should be used on TS which are very hard to model, highly irregular, for which it is difficult to extract any information and when constraints for time and computational resources are not critical.

Also a natural succeeding step is to use well performing GP within WA-based forecasting algorithm. Using GP for extrapolating each wavelet level could improve the final result compared to both typical WA and pure GP approach. Additionally, there should be no dramatic extension to time of calculations as, due to the properties of wavelets, calculations of each wavelet level could be processed separately, on different machines. However, this parallelization does not reduce the total cost of calculations, it only reduces the time in which those could be performed. This approach is currently being examined and preliminary results reveal it as a promising direction for further research.

References

1. Aczel, A. D.: Complete business statistics. Boston, MA: Irwin. (1989)
2. Biethahn, J., Nissen, V.: Evolutionary algorithms in management applications. Berlin, Germany: Springer-Verlag. (1995)
3. Chen, S. H.: Evolutionary computation in economics and finance. Heidelberg, New York: Physica-Verlag. (2002)
4. Deb, K. e. a.: Genetic evolutionary computation - GECCO 2004 (Vol. 3102). Berlin, Germany: Springer-Verlag. (2004)
5. Donoho, D. L., Johnstone, I. M.: Adapting to unknown smoothness via wavelet shrinkage. *Journal of American Statistical Association* **90** (1995) 1200–1224.
6. Goldberg, D.: Genetic algorithms in search, optimization and machine learning. Reading, MA: Addison-Wesley. (1989)
7. Haar, A.: Zur Theorie der orthogonalen Funktionensysteme. *Mathematische Annalen* **LXIX** (1910) 331–371.
8. Koza, J. R.: Genetic programming: On the programming of computers by means of natural selection. Cambridge, MA: The MIT Press. (1992)
9. Kozłowski, B.: On time series forecasting methods of linear complexity utilizing wavelets. In: *Advances in Intelligent Systems - Theory and Applications*, In Cooperation with the IEEE Computer Society. Kirchberg - Luxembourg. (2004)
10. Kozłowski, B.: Time series denoising with wavelet transform. *Journal of Telecommunications and Information Technology* **3** (2005) 91–95.
11. Li, T., Li, Q., Zhu, S., Ogihara, M.: Survey on wavelet applications in data mining. *SIGKDD EXplorations* **4** (2003) 49–68.
12. Michalewicz, Z.: Genetic algorithms + data structures = evolution programs. Berlin, Germany: Springer-Verlag. (1996)

13. Murphy, J. J.: Technical analysis of futures markets: A comprehensive guide to trading methods and applications. New York, NY: New York Inst. of Finance. (1986)
14. Percival, D. B., Walden, A. T.: Wavelet methods for time series analysis. Cambridge, UK: Cambridge University Press. (2000)
15. Prasad, L., Iyengar, S. S.: Wavelet analysis with applications to image processing. USA: CRC Press. (1997)
16. Rhea, R.: The dow theory. USA: Fraser Publishing. (1993)

Gradient Based Stochastic Mutation Operators in Evolutionary Multi-objective Optimization

Pradyumn Kumar Shukla

Institute of Numerical Mathematics, Department of Mathematics,
Dresden University of Technology, Dresden PIN 01069, Germany
pradyumn.shukla@mailbox.tu-dresden.de

Abstract. Evolutionary algorithms have been adequately applied in solving single and multi-objective optimization problems. In the single-objective case various studies have shown the usefulness of combining gradient based classical search principles with evolutionary algorithms. However there seems to be a dearth of such studies for the multi-objective case. In this paper, we take two classical search operators and discuss their use as a local search operator in a state-of-the-art evolutionary algorithm. These operators require gradient information which is obtained using a stochastic perturbation technique requiring only two function evaluations. Computational studies on a number of test problems of varying complexity demonstrate the efficiency of hybrid algorithms in solving a large class of complex multi-objective optimization problems.

1 Introduction

Multi-objective optimization is a rapidly growing area of research and application in modern-day optimization. There exist a plethora of non-classical methods which follow some natural or physical principles for solving multi-objective optimization problems, see for example the book by [4]. On the other hand a large amount of studies have been devoted to develop classical methods for solving multi-objective optimization problems.

Evolutionary algorithms use stochastic transition rules using crossover and mutation search operators to move from one solution to another. In this way global structure of search space is exploited. Classical methods, on the other hand, usually use deterministic (usually gradient based) transition rules to move from one solution to another. Classical methods effectively use local information thus ensuring fast convergence. This however comes up at the cost of requiring gradient or Hessian information which requires a large number of function evaluations. Hence one sees that there is a trade-off between fast convergence and number of function evaluations. Hybrid implementations thus continue to be developed and tested (see for example [18,9,23]).

This study is motivated by the an earlier comparative study [14]. In this contribution we take two classical gradient based Pareto front generating methods and use their search principles as mutation operators in a state-of-the-art multi-objective evolutionary algorithm to create a powerful hybrid multi-objective

metaheuristics algorithm. We demonstrate their efficiency in solving real valued problems of varying complexity.

This paper is structured as follows. The next section present an overview of various classical generating methods and the gradient estimation technique, while the third section presents the simulation results. Conclusions as well as extensions which emanated from this study are presented in the end of this contribution.

2 Classical Search Principles

For the present study we take two classical algorithms and use their search operators as mutation operators in the elitist non-dominated sorting GA or NSGA-II developed by [5]. The gradients of objective functions are obtained by a stochastic method described later in this section. These classical algorithms and their search operators are described next.

2.1 Schäffler's Stochastic Method (SSM)

This method [13], is based on the solution of a set of stochastic differential equations. It method requires the objective functions to be twice continuously-differentiable. In each iteration, a trace of non-dominated points is constructed by calculating at each point \mathbf{x} a direction $(-q(\mathbf{x}))$ in the decision space which is a direction of descent for *all* objective functions (note that we consider m to be the number of objective functions denoted by f_i for all $i = 1, 2, \dots, m$). The direction of descent is obtained by solving a quadratic subproblem. Let $\hat{\alpha}$ be the minimizer. Then $q(\mathbf{x}) = \sum_{i=1}^m \hat{\alpha}_i \nabla f_i(\mathbf{x})$. A set of non-dominated solution is obtained by perturbing the solution (minimum along the direction of descent) using a Brownian motion concept. The following stochastic differential equation (SDE) is employed for this purpose:

$$d\mathbf{X}_t = -q(\mathbf{X}_t)d(t) + \varepsilon dB_t, \quad \mathbf{X}_0 = \mathbf{x}_0, \quad (1)$$

where $\varepsilon > 0$ and B_t is a n -dimensional Brownian motion. As the first search operator we use Equation 1 to create a child instead of the mutation operator. The gradients are obtained using a stochastic perturbation method described later. We name the hybrid algorithm with new mutation operator as S-NSGA-II. In all simulations here, to solve the above equation numerically, we employ the Euler's method with a step size σ . The approach needs two parameters to be set properly: (i) the parameter ε which controls the amount of global search and (ii) the step size σ used in the Euler's approach which controls the accuracy of the integration procedure.

2.2 Timmel's Population Based Method (TPM)

As early as in 1980, [16,17] proposed a population-based stochastic approach for finding multiple Pareto-optimal solutions of a differentiable multi-objective

optimization problem. In this method, first, a feasible solution set (we call it a population) is randomly created. The non-dominated solutions ($\mathbf{X}_0 = \{\mathbf{x}_1^0, \mathbf{x}_2^0, \dots, \mathbf{x}_s^0\}$) are identified and they serve as the first approximation to the Pareto-optimal set. Thereafter, from each solution \mathbf{x}_k^0 , a child solution is created in the following manner:

$$\mathbf{x}_k^1 = - \sum_{i=1}^M t_1 u_i \nabla f_i(\mathbf{x}_k^0), \quad (2)$$

where u_i is the realization of a uniformly distributed random number (between 0 and 1) and t_1 is step-length in the first generation. It is a simple exercise to show that the above formulation ensures that not all functions can be worsened simultaneously. The variation of the step-length over iterations must be made carefully to ensure convergence to the efficient frontier. The original study suggested the following strategy for varying the step-length t_j with generation j : $t_j = C/j$ (where C is a positive constant). As the second search operator we use Equation 2 to create a child instead of the mutation operator. As in S-NSGA-II the gradients are obtained using a stochastic perturbation method described later. We name the hybrid algorithm with new mutation operator as T-NSGA-II.

2.3 Gradient Estimation: Simultaneous Perturbation Method

In almost all classical algorithms (for both single and multi-objective problems) the gradient of a function (say in general h) are required. The standard approach for estimating the gradient is the Finite Difference (FD) method. For variable (say \mathbf{x}) of dimension n this method of gradient estimation requires $2n$ function evaluations. This is costly in terms of function evaluations (of the order $O(n)$). The Simultaneous Perturbation (SP) method [15] on the other hand requires only *two* function evaluation independently of n (computational complexity is thus $O(1)$) as follows

$$g_i(\mathbf{x}) = \frac{f(x + c\Delta) - f(x - c\Delta)}{2c\Delta_i},$$

where the i^{th} component of the gradient is denoted by $g_i(\mathbf{x})$, Δ is a n dimensional vector of random perturbations satisfying certain statistical conditions ([15]). A simple (and theoretically valid) choice for each component of Δ is to use a Bernoulli distribution ± 1 with probability of 0.5 for each ± 1 outcome. The step size c at each iteration (denoted by c_k) is given as $c_k = c_0/(k+1)^\gamma$. Practically effective (and theoretically valid [15]) values of c_0, γ are 0.001 and $1/6$ which are used here.

3 Simulation Results

In this section, we compare the above two hybrid methods with the elitist non-dominated sorting GA or NSGA-II [5] on a number of test problems. The test problems are chosen in such a way so as to systematically investigate various

aspects of an algorithm. For S-NSGA-II the parameters $\sigma = 1.0$ along with $\epsilon = 0.1$ is used for all the test problems. For T-NSGA-II the parameter $C = 10.0$ is used (unless otherwise stated). For the NSGA-II, we use a standard real-parameter SBX and polynomial mutation operator with $\eta_c = 10$ and $\eta_m = 10$, respectively [4] (unless otherwise stated). For all problems solved, we use a population of size 100. We set the number of function evaluations as 5000 for each problems and 15000 for difficult ones.

Convergence and diversity are two distinct goals in multi-objective optimization. In order to evaluate convergence we use the Generational Distance (GD) metric [4]. Diversity of solutions is evaluated using the Spread (denoted by S) metric [4]. These unary metrics for convergence and diversity are used together with a binary metric which can detect whether an approximation set is better than another. We use the multiplicative binary ϵ indicator discussed by Zitzler [18] to assess the performance of the algorithms. Also for statistical evaluation we use attainment surface based statistical metric ([7]) for one hard problem. We consider two-objective ZDT test problems discussed in [4]. The test problems are slightly modified so that they become unconstrained multi-objective optimization problems, as the SSM method is only able to tackle unconstrained problems in its present form. Table 1 present these test problems. Also the box constraints are slightly modified so that the functions are twice continuously differentiable in the entire feasible region (required as per SSM).

Table 1. Test problems

| Name | Objectives | g | Domain |
|------|---|---|---------------------------------|
| ZDT1 | $f_1(\mathbf{x}) = x_1, f_2(\mathbf{x}) = g(\mathbf{x}) \left(2 - \sqrt{\frac{x_1}{g(\mathbf{x})}} \right)$ | $g(x) = 1 + \frac{9}{n-1} \sum_{i=2}^n x_i^2$ | $[0.01, 1] \times [-1, 1]^{29}$ |
| ZDT2 | $f_1(\mathbf{x}) = x_1, f_2(\mathbf{x}) = g(\mathbf{x}) \left(2 - \left(\frac{x_1}{g(\mathbf{x})} \right)^2 \right)$ | $g(x) = 1 + \frac{9}{n-1} \sum_{i=2}^n x_i^2$ | $[0.01, 1] \times [-1, 1]^{29}$ |
| ZDT3 | $f_1(\mathbf{x}) = x_1, f_2(\mathbf{x}) = g(x) \left(2 - \sqrt{\frac{x_1}{g(\mathbf{x})}} - \frac{x_1}{g(\mathbf{x})} \sin(10\pi x_1) \right)$ | $g(x) = 1 + \frac{9}{n-1} \sum_{i=2}^n x_i^2$ | $[0.01, 1] \times [-1, 1]^{29}$ |
| ZDT4 | $f_1(\mathbf{x}) = x_1, f_2(\mathbf{x}) = g(\mathbf{x}) \left(2 - \sqrt{\frac{x_1}{g(\mathbf{x})}} \right)$ | $g(x) = 1 + 10(n-1) + \sum_{i=2}^n (x_i^2 - 10 \cos(4\pi x_i))$ | $[0.01, 1] \times [-5, 5]^{19}$ |
| ZDT6 | $f_1(\mathbf{x}) = 1 - \exp(-4x_1) \sin^6(4\pi x_1), f_2(\mathbf{x}) = g(\mathbf{x}) \left(2 - \left(\frac{f_1(\mathbf{x})}{g(\mathbf{x})} \right)^2 \right)$ | $g(x) = 1 + 9 \left(\frac{1}{n-1} \sum_{i=2}^n x_i^2 \right)^{0.25}$ | $[0.01, 1] \times [-1, 1]^9$ |

The modified ZDT1 problem has a convex Pareto-optimal front for which solutions correspond to $0.01 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, 3, \dots, 30$. Figure 1 shows the performance of all the algorithms after 5000 function evaluations. Table 2 shows the binary ϵ indicator values of all the algorithms on this problem. An element (i, j) in this table represents $I_\epsilon(\text{algorithm } j, \text{algorithm } i)$. Given two outcomes A and B, of different algorithms, the binary ϵ indicator $I_\epsilon(A, B)$ gives the factor by which an approximation set is worse than another with respect

to all objectives. If $I_\epsilon(A, B) \leq 1$ and $I_\epsilon(B, A) > 1$ occurs then we can conclude that Algorithm A is better than Algorithm B. These conditions are quite difficult to satisfy using binary ϵ indicator values. We will use the binary ϵ indicator values to conclude *partial* results: we will say that Algorithm A is *relatively better* than Algorithm B if $I_\epsilon(A, B) \leq 1.05$ and $I_\epsilon(B, A) > 1.05$. If $I_\epsilon(A, B) \leq 1$ and $I_\epsilon(B, A) > 1$ occurs then we say that Algorithm A is *definitively better* than Algorithm B. From Table 2 we obtain that with respect to binary ϵ metric all the three algorithms are incomparable. However, with respect to diversity S-NSGA-II performs the best (Table 4) while T-NSGA-II performs best in terms of convergence (Table 5).

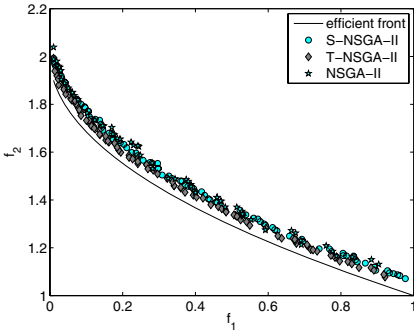


Fig. 1. Performance of the three algorithms on ZDT1

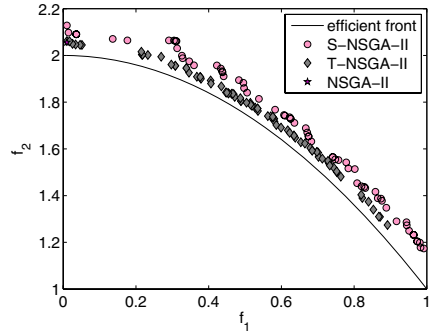


Fig. 2. Performance of the three algorithms on ZDT2

The modified ZDT2 problem has a non-convex Pareto-optimal front for which solutions correspond to $0.01 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, 3, \dots, 30$. In the case of TPM algorithm (using a limited parametric study) we choose $C = 4.0$. Figure 2 shows the performance of all the algorithms after 5000 function evaluations. It can be seen both T-NSGA-II and S-NSGA-II find much more non-dominated solutions close to efficient front than NSGA-II. Although from Table 3 we observe that NSGA-II is *relatively better* than other two algorithms, however this happens since NSGA-II find *one* solution close to efficient front (while S-NSGA-II and T-NSGA-II find 38 and 37 solutions respectively). As can be seen from tables 4 and 5 that spread of both S-NSGA-II and T-NSGA-II is better than NSGA-II while convergence of T-NSGA-II is best. Next we consider ZDT3, this problem

Table 2. Binary ϵ indicator values on ZDT1

| | SSM | TPM | NSGA |
|-----------|--------|--------|--------|
| S-NSGA-II | 1.0000 | 1.0124 | 1.0558 |
| T-NSGA-II | 1.0374 | 1.0600 | 1.0494 |
| NSGA-II | 1.0160 | 1.0056 | 1.0000 |

Table 3. Binary ϵ indicator values on ZDT2

| | SSM | TPM | NSGA |
|-----------|--------|--------|--------|
| S-NSGA-II | 1.0000 | 1.0854 | 1.7517 |
| T-NSGA-II | 1.0512 | 1.0000 | 1.6139 |
| NSGA-II | 1.0353 | 1.0027 | 1.0000 |

has a convex discontinuous efficient frontier. Figure 3 shows the performance of all the algorithms after 5000 function evaluations. It can be seen that all the algorithms find non-dominated solutions close to efficient front in its all the 5 disconnected parts. From Table 6 one infers that S-NSGA-II performs relatively better than original NSGA-II, while T-NSGA-II and NSGA-II are incomparable. As can be seen from tables 4 and 5 diversity of T-NSGA-II is the best, while convergence of S-NSGA-II is best.

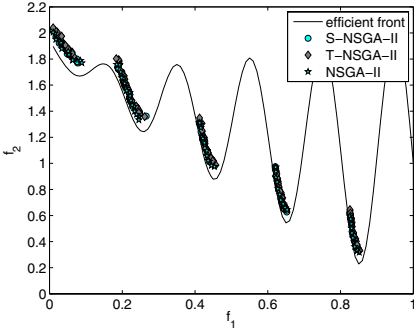


Fig. 3. Performance of the three algorithms on ZDT3

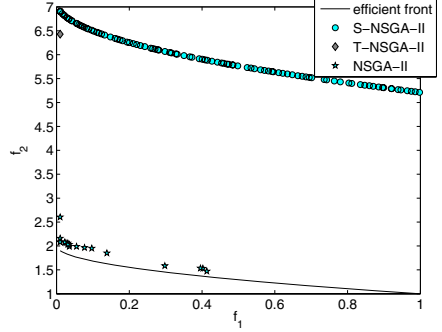


Fig. 4. Performance of the three algorithms on ZDT4

Table 4. Spread metric values. Best values are shown in bold.

| | ZDT1 | ZDT2 | ZDT3 | ZDT4 | ZDT6 |
|-----------|---------------|---------------|---------------|---------------|---------------|
| S-NSGA-II | 0.4992 | 0.7299 | 0.8419 | 0.8900 | 0.5084 |
| T-NSGA-II | 0.6002 | 0.8018 | 0.7914 | 1.0000 | 0.3799 |
| NSGA-II | 0.7196 | 1.000 | 0.7998 | 1.0817 | 0.6267 |

The problem ZDT4 has a total of 100 distinct local efficient fronts in the objective space. The global Pareto-optimal solutions correspond to $0.01 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, 3, \dots, 10$. Since ZDT4 is a complex multi-modal problem in this problem all the algorithms are run till 15000 function evaluations. Figure 4 shows the performance of all the algorithms. It can be seen that only NSGA-II is able to overcome many local Pareto optimal fronts and also performs best in terms of convergence (Table 5).

Next we consider another difficult problem, ZDT6. This problem has a non-convex and non-uniformly spaced Pareto-optimal solutions. The Pareto-optimal solutions correspond to $0.01 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, 3, \dots, 10$. In case of NSGA-II we use $\eta_c = 1$ and $\eta_m = 1$ as these are more efficient for such difficult problems. Similarly, in the case of T-NSGA-II algorithm we choose $c = 0.5$ while in case of S-NSGA-II we use the same parameters. ZDT6 is a hard multi-objective problems and thus we run the simulations for a total of 15000 generations for each algorithm. Figure 5 shows the performance of all the algorithms after 15000

Table 5. Generational distance metric values. Best values are shown in bold.

| | ZDT1 | ZDT2 | ZDT3 | ZDT4 | ZDT6 |
|-----------|---------------|---------------|---------------|---------------|---------------|
| S-NSGA-II | 0.0521 | 0.0825 | 0.0267 | 4.1118 | 0.0354 |
| T-NSGA-II | 0.0342 | 0.0407 | 0.0388 | 4.5287 | 0.0214 |
| NSGA-II | 0.0584 | 0.0562 | 0.0283 | 0.1673 | 0.1864 |

Table 6. Binary ϵ indicator values on ZDT3

| | SSM | TPM | NSGA |
|-----------|--------|--------|---------|
| S-NSGA-II | 1.0000 | 1.0454 | 1.02810 |
| T-NSGA-II | 1.0160 | 1.0000 | 1.0113 |
| NSGA-II | 1.0620 | 1.0452 | 1.0000 |

Table 7. Binary ϵ indicator values on ZDT6

| | SSM | TPM | NSGA |
|-----------|--------|--------|--------|
| S-NSGA-II | 1.0000 | 1.0028 | 1.2412 |
| T-NSGA-II | 1.0310 | 1.0000 | 1.2797 |
| NSGA-II | 1.0000 | 1.0000 | 1.0000 |

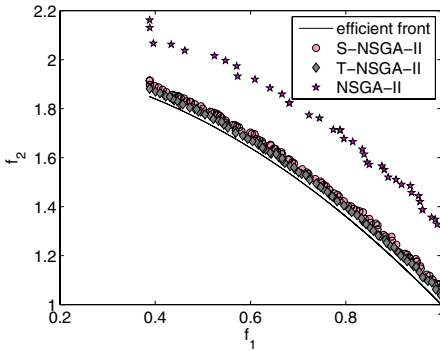


Fig. 5. Performance of the three algorithms on ZDT6

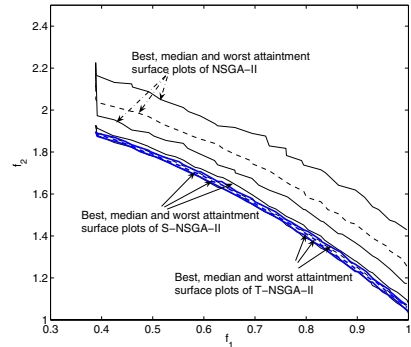


Fig. 6. Attainment surface plots of all the algorithms

function evaluations. It can be easily seen that both S-NSGA-II and T-NSGA-II perform better than NSGA-II. Also from Table 8 it can be inferred that performance of both S-NSGA-II and T-NSGA-II is *definitively better* than NSGA-II. The algorithms S-NSGA-II and T-NSGA-II are also better than NSGA-II in terms of diversity (Table 4) and convergence (Table 5). In order to address the issue of stochasticity on this hard problem we perform 21 different runs of all the algorithms and plot the best, median and worst attainment surface plot. Figure 6 shows the plots. One observes that even the worst attainment surfaces of both S-NSGA-II and T-NSGA-II are much better than that of NSGA-II.

4 Conclusions

On a number of test problems of varying complexity, it has been observed that using the TPM and SSM as mutation search operators in NSGA-II performs

better than the standard NSGA-II approach for a wide class of problems. However, for problems having multi-modal efficient front the above classical search techniques have not performed well. However this does not mean that they are not suited for solving multi-modal problems. Take for example, TPM in this the parameter C is crucial to get to the global efficient front. Once TPM gets stuck one strategy could be to restart the series by taking some larger value of C . SSM on the hand has a global search operator built in that and thus better numerical methods of solving the stochastic differential equation should be taken to see its performance. S-NSGA-II and T-NSGA-II perform substantially well over the evolutionary method in case of hard problems having a non-uniform density of solutions in the objective space. On the other hand, on all other problems considered here, the S-NSGA-II, has performed well in achieving both convergence and diversity of solutions with the *same* parameter values in all the test problems.

As an extension to this study one could use classical techniques for box-constrained [12], nonlinear inequality constrained [11,6], and non-differentiable [10] problems and try to combine with evolutionary optimization to create powerful hybrid algorithms.

Acknowledgements. The author acknowledges the partial financial support by the Gottlieb-Daimler- and Karl Benz-Foundation.

References

1. Peter A. N. Bosman and Edwin D. de Jong. Exploiting gradient information in numerical multi-objective evolutionary optimization. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 755–762, New York, NY, USA, 2005. ACM Press.
2. Peter A. N. Bosman and Edwin D. de Jong. Combining gradient techniques for numerical multi-objective evolutionary optimization. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 627–634, New York, NY, USA, 2006. ACM Press.
3. Martin Brown and Robert E. Smith. Effective use of directional information in multi-objective evolutionary computation. In *GECCO '03: Proceedings of the 5th annual conference on Genetic and evolutionary computation*, pages 778–789, 2003.
4. K. Deb. *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley, 2001.
5. K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
6. Jörg Fliege and Benar Fux Svaiter. Steepest descent methods for multicriteria optimization. *Math. Methods Oper. Res.*, 51(3):479–494, 2000.
7. C. M. Fonesca and P. J. Fleming. On the performance assessment and comparison of stochastic multiobjective optimizers. In *Proceedings of Parallel Problem Solving from Nature IV (PPSN-IV)*, pages 584–593, 1996.
8. Ken Harada, Kokolo Ikeda, and Shigenobu Kobayashi. Hybridization of genetic algorithm and local search in multiobjective function optimization: recommendation

- of ga then ls. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 667–674, New York, NY, USA, 2006. ACM Press.
9. Ken Harada, Jun Sakuma, and Shigenobu Kobayashi. Local search for multiobjective function optimization: pareto descent method. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 659–666, New York, NY, USA, 2006. ACM Press.
 10. K. C. Kiwiel. Descent methods for nonsmooth convex constrained minimization. In *Nondifferentiable optimization: motivations and applications (Sopron, 1984)*, volume 255 of *Lecture Notes in Econom. and Math. Systems*, pages 203–214. Springer, Berlin, 1985.
 11. H. Mukai. Algorithms for multicriterion optimization. *IEEE Trans. Automat. Control*, 25(2):177–186, 1980.
 12. Maria Cristina Recchioni. A path following method for box-constrained multiobjective optimization with applications to goal programming problems. *Math. Methods Oper. Res.*, 58(1):69–85, 2003.
 13. S. Schäffler, R. Schultz, and K. Weinzierl. Stochastic method for the solution of unconstrained vector optimization problems. *Journal of Optimization Theory and Applications*, 114(1):209–222, 2002.
 14. P. K. Shukla, K. Deb, and S. Tiwari. Comparing Classical Generating Methods with an Evolutionary Multi-objective Optimization Method. In C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, editors, *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*, pages 311–325, Guanajuato, México, March 2005. Springer. Lecture Notes in Computer Science Vol. 3410.
 15. J. C. Spall. Implementation of the Simultaneous Perturbation Algorithm for Stochastic Optimization. *IEEE Transactions on Aerospace and Electronic Systems*, 34(3):817–823, 1998.
 16. G. Timmel. Ein stochastisches Suchverfahren zur Bestimmung der optimalen Kompromissungen bei statischen polzkriteriellen Optimierungsaufgaben. *Wiss. Z. TH Ilmenau*, 26(5):159–174, 1980.
 17. G. Timmel. Modifikation eines statistischen Suchverfahrens der Vektoroptimierung. *Wiss. Z. TH Ilmenau*, 28(6):139–148, 1982.
 18. Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M. Fonseca, and Viviane Grunert da Fonseca. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, April 2003.

Co-evolutionary Multi-agent System with Predator-Prey Mechanism for Multi-objective Optimization

Rafał Drezewski and Leszek Siwik

Department of Computer Science
AGH University of Science and Technology, Kraków, Poland
{drezew, siwik}@agh.edu.pl

Abstract. Co-evolutionary techniques for evolutionary algorithms allow for the application of such algorithms to problems for which it is difficult or even impossible to formulate explicit fitness function. These techniques also maintain population diversity, allows for speciation and help overcoming limited adaptive capabilities of evolutionary algorithms. In this paper the idea of *co-evolutionary multi-agent system with predator-prey mechanism for multi-objective optimization* is introduced. In presented system the Pareto frontier is located by the population of agents as a result of co-evolutionary interactions between two species: predators and prey. Results from runs of presented system against test problem and comparison to classical multi-objective evolutionary algorithms conclude the paper.

1 Introduction

Co-evolution is the biological process responsible for speciation, maintaining population diversity, introducing arms races and open-ended evolution. In *co-evolutionary algorithms (CoEAs)* the fitness of each individual depends not only on the quality of solution to the given problem (like in *evolutionary algorithms (EAs)*) but also (or solely) on other individuals' fitness [7]. This makes such techniques applicable in the cases where the fitness function formulation is difficult (or even impossible). Co-evolutionary techniques for EAs are also aimed at improving adaptive capabilities in dynamic environments and introducing open-ended evolution and speciation into EAs by maintaining population diversity. As the result of ongoing research quite many co-evolutionary techniques have been proposed. Generally, each of these techniques belongs to one of two classes: competitive or cooperative.

Optimization problems in which multiple criteria have to be taken into account are called multi-objective (or multi-criteria) problems [2,10]. Most real-life decision processes are such problems—decision maker must deal with multiple criteria (objectives). A solution in the Pareto sense of the multi-objective optimization problem means determination of all non-dominated (in the sense of *weak domination relation* [10]) alternatives from the set of all possible (feasible) decision alternatives. The set of all non-dominated alternatives is sometimes called a *Pareto-optimal set*. These locally or globally non-dominated solutions create (in the criteria space) so-called local or global Pareto frontiers [2].

In the recent years evolutionary approach to multi-objective problems is the subject of intensive research. As a result of this research a variety of evolutionary multi-objective optimization techniques have been proposed. Evolutionary multi-objective algorithms (EMOAs) can be generally classified as elitist (in which best individuals can be directly carried over to the next generation) and non-elitist ones [2].

In the case of multi-objective optimization—high quality approximation of *Pareto frontier* should fulfill at least three distinguishing features: first of all it should be “located” as close to the ideal Pareto frontier as possible, secondly it should include as many alternatives as possible, and all proposed non-dominated alternatives should be evenly distributed over the whole ideal Pareto set.

In the case of multi-objective optimization premature loss of population diversity can result not only in lack of drifting to the ideal Pareto frontier but also in obtaining approximation of Pareto set that is focused around its selected area(s)—what of course is very undesirable assuming that preference-based multi-objective optimization is not considered in this place. Additionally, in the case of multi-objective problems with many local Pareto frontiers (so called multi-modal multi-objective problems defined by Deb in [2]) the loss of population diversity may result in locating only local Pareto frontier instead of a global one.

The basic idea of *evolutionary multi-agent systems (EMAS)* is the realization of evolutionary processes within the confines of multi-agent system. EMAS systems have already been applied successfully to discrete, continuous, combinatorial and non-combinatorial multi-objective optimization problems ([8]). It has been also shown that on the basis of the EMAS further research and more sophisticated approaches, models and mechanisms can be proposed [9].

The model of *co-evolutionary multi-agent system (CoEMAS)*, as opposed to the basic *evolutionary multi-agent system (EMAS)* model, allows for the co-existence of several species and sexes which can interact with each other and co-evolve. Co-evolutionary mechanisms can serve as the basis for niching and speciation techniques for EMAS systems [3]. CoEMAS systems can also be applied to multi-objective optimization, especially when there is need for maintaining population diversity and speciation [4,5].

In the following sections the formal model of co-evolutionary multi-agent system with predator-prey mechanism is presented. Such system is applied to multi-objective optimization Kursawe problem and compared to other selected classical evolutionary techniques.

2 Co-evolutionary Multi-agent System with Predator-Prey Mechanism for Multi-objective Optimization

The system presented in this paper is the CoEMAS with predator-prey mechanism (see fig. 1). There are two species: predators and prey in this system. Prey represent solutions of the multi-objective problem. The main goal of predators is to eliminate “weak” (i.e. dominated) prey.

The *CoEMAS* is described as 4-tuple: $CoEMAS = \langle E, S, \Gamma, \Omega \rangle$ where E is the environment of the *CoEMAS*, S is the set of species ($s \in S$) that co-evolve in *CoEMAS*, Γ is the set of resource types that exist in the system, the amount of type γ resource will be

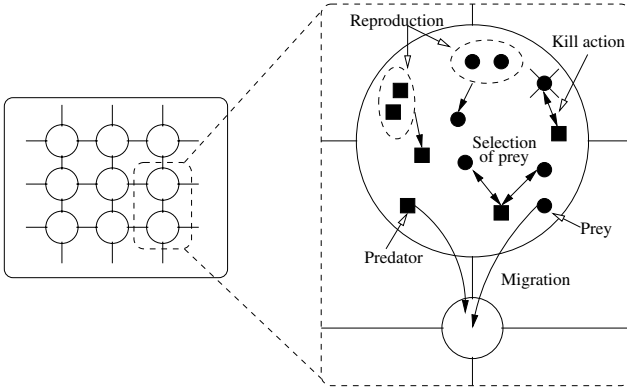


Fig. 1. CoEMAS with predator-prey mechanism

denoted by r^γ , Ω is the set of information types that exist in the system, the information of type ω will be denoted by i^ω . There are two information types ($\Omega = \{\omega_1, \omega_2\}$) and one resource type ($\Gamma = \{\gamma\}$) in *CoEMAS*. Informations of type ω_1 contain nodes to which agent can migrate, when it is located in particular node of the graph. Informations of type ω_2 contain agents-prey which are located in the particular node in time t . There is one resource type ($\Gamma = \{\gamma\}$) in *CoEMAS*, and there is closed circulation of resource within the system.

The **environment** of *CoEMAS* is defined as 3-tuple: $E = \langle T^E, \Gamma^E = \Gamma, \Omega^E = \Omega \rangle$, where T^E is the topography of environment E , Γ^E is the set of resource types that exist in the environment, Ω^E is the set of information types that exist in the environment. The topography of the environment $T^E = \langle H, l \rangle$ where H is directed graph with the cost function c defined ($H = \langle V, B, c \rangle$, V is the set of vertices, B is the set of arches). The distance between two nodes is defined as the length of the shortest path between them in graph H . The $l: A \rightarrow V$ (A is the set of agents, that exist in *CoEMAS*) function makes it possible to locate particular agent in the environment space.

Vertice v is given by: $v = \langle A^v, \Gamma^v = \Gamma^E, \Omega^v = \Omega^E, \varphi \rangle$, where A^v is the set of agents that are located in the vertice v . Agents can collect two types of informations from the vertice. The first one includes all vertices that are connected with the vertice v and the second one includes all agents of species *prey* that are located in the vertice v .

The **set of species** is given by: $S = \{prey, pred\}$. The prey species is defined as follows: $prey = \langle A^{prey}, SX^{prey} = \{sx\}, Z^{prey}, C^{prey} \rangle$, where A^{prey} is the set of agents of prey species, SX^{prey} is the set of sexes which exist within the *prey* species, Z^{prey} is the set of actions that agents of species *prey* can perform, and C^{prey} is the set of relations of species *prey* with other species that exist in the *CoEMAS*. There is only one sex sx ($sx \equiv sx^{prey}$) within the *prey* species, which is defined as follows: $sx = \langle A^{sx} = A^{prey}, Z^{sx} = Z^{prey}, C^{sx} = \emptyset \rangle$.

The set of actions $Z^{prey} = \{die, get, give, accept, seek, clone, rec, mut, migr\}$, where *die* is the action of death (prey dies when it is out of resources), *get* action gets some resource from another a^{prey} agent located in the same node (this agent is dominated by the agent that performs *get* action or is too close to him in the criteria space — such agent

is found with the use of *seek* action), *give* actions gives some resource to another agent (which performs *get* action), *accept* action accepts partner for reproduction (partner is accepted when the amount of resource possessed by the prey agent is above the given level), *seek* action seeks for another prey agent that is dominated by the prey performing this action (or too close to it in criteria space) or seeks for partner for reproduction (prey starts seeking partner for reproduction when the amount of resource is above the given level), *clone* is the action of cloning prey (new agent with the same genotype as parent's one is created), *rec* is the recombination operator (intermediate recombination is used [11]), *mut* is the mutation operator (mutation with self-adaptation is used [11]), *migr* action allows prey to migrate between nodes of graph H (migrating agent loses some resource).

The set of relations of *prey* species with other species that exist within the system is defined as follows: $C^{prey} = \left\{ \xrightarrow{prey.get^-} = \{\langle prey, prey \rangle\}, \xrightarrow{pred.give^+} = \{\langle prey, pred \rangle\} \right\}$. The first relation models intra species competition for limited resources (as a result of performing *get* action the fitness of another prey is decreased — “-”). The second one models predator-prey interactions (prey gives all the resource it owns to predator and dies — predator fitness is increased: “+”).

The **predator species** (*pred*) is defined analogically as *prey* species with the following differences. The set of actions $Z^{pred} = \{seek, get, migr\}$, where *seek* action seeks for the “worst” (according to the criteria associated with the given predator) prey located in the same node as predator performing this action, *get* action gets all resource from chosen prey, *migr* action allows predator to migrate between nodes of graph H (migrating agent loses some resource — if it can not afford the migration it stays at the same node).

The set of relations of *pred* species with other species that exist within the system is defined as follows: $C^{pred} = \left\{ \xrightarrow{prey.get^-} = \{\langle pred, prey \rangle\} \right\}$. The relation models predator-prey interactions (predator gets all resources from selected prey, decreases its fitness and prey dies).

Agent a of species *prey* is given by: $a = \langle gn^a, Z^a = Z^{prey}, \Gamma^a = \Gamma, \Omega^a = \Omega, PR^a \rangle$. Genotype of agent a is consisted of two vectors (chromosomes): x of real-coded decision parameters' values and σ of standard deviations' values, which are used during mutation. $Z^a = Z^{prey}$ is the set of actions which agent a can perform. Γ^a is the set of resource types, and Ω is the set of information types.

The set of profiles PR^a includes resource profile (pr_1), reproduction profile (pr_2), interaction profile (pr_3), and migration profile (pr_4). Each time step prey tries to realize goals of the profiles (taking into account the priorities of the profiles: $pr_1 \preceq pr_2 \preceq pr_3 \preceq pr_4$ — here pr_1 has the highest priority). In order to realize goals of the given profile agent uses strategies which can be realized within this profile.

Within pr_1 profile all strategies connected with type γ resource are realized ($\langle die \rangle$, $\langle seek, get \rangle$). This profile uses informations of type ω_2 . Within pr_2 profile strategy of reproduction ($\langle seek, clone, rec, mut \rangle$) is realized (informations of type ω_2 are used and reproducing prey give some resource to child with the use of *give* action). Within pr_3 profile the interactions with predators are realized (strategy $\langle give \rangle$). Within pr_4 profile

Table 1. Comparison of proposed CoEMAS approach with selected classical EMOA's according to the *HV* and *HVR* metrics obtained during solving Kursawe problem

| HV / HVR | | | |
|----------|----------------|----------------|-----------------|
| Step | CoEMAS | PPES | NPGA |
| 1 | 541.21 / 0.874 | 530.76 / 0.857 | 489.34 / 0.790 |
| 10 | 588.38 / 0.950 | 530.76 / 0.867 | 563.55 / 0.910 |
| 20 | 594.09 / 0.959 | 531.41 / 0.858 | 401.79 / 0.648 |
| 30 | 601.66 / 0.971 | 531.41 / 0.858 | 378.78 / 0.611 |
| 40 | 602.55 / 0.973 | 531.41 / 0.858 | 378.73 / 0.611 |
| 50 | 594.09 / 0.959 | 531.41 / 0.858 | 378.77 / 0.611 |
| 100 | 603.04 / 0.974 | 531.42 / 0.858 | 378.80 / 0.6117 |
| 600 | 603.79 / 0.975 | 577.44 / 0.932 | 378.80 / 0.611 |
| 200 | 611.43 / 0.987 | 609.47 / 0.984 | 378.80 / 0.611 |
| 4000 | 611.44 / 0.987 | 555.53 / 0.897 | 378.80 / 0.611 |
| 6000 | 613.10 / 0.990 | 547.73 / 0.884 | 378.80 / 0.611 |

the migration strategy ($\langle migr \rangle$), which uses information i^{ω_1} , is realized—as a result of performing this strategy prey loses some resource.

Agent a of species *pred* is defined analogically to *prey* agent. The main differences are genotype and the set of profiles. Genotype of agent a is consisted of the information about the criterion associated with this agent. The set of profiles PR^a includes resource profile (pr_1), and migration profile (pr_2). Within pr_1 profile all strategies connected with type γ resource are realized ($\langle seek, get \rangle$). This profile uses informations of type ω_2 . Within pr_2 profile the migration strategy ($\langle migr \rangle$), which uses information i^{ω_1} , is realized. As a result of performing this strategy predator loses some resource.

3 Test Problem and Experimental Results

Presented in the course of this paper agent-based co-evolutionary approach for multi-objective optimization has been tested using a lot of benchmark problems such as Kursawe problem, Laumanns problem, set of Zitzler's problems etc. Because of space limitations it is possible to present in this paper only selected results. Authors decided to discuss obtained results on the basis of Kursawe problem. Its definition is as follows:

$$Kursawe = \begin{cases} f_1(x) = \sum_{i=0}^{n-1} \left(-10 \exp \left(-0.2 \sqrt{x_i^2 + x_{i+1}^2} \right) \right) \\ f_2(x) = \sum_{i=1}^n |x_i|^{0.8} + 5 \sin x_i^3 \\ n = 3 \quad -5 \leq x_1, x_2, x_3 \leq 5 \end{cases}$$

In the case of Kursawe problem optimization algorithm has to deal with disconnected two-dimensional Pareto frontier and disconnected three dimensional Pareto set. Additionally, a specific definition of f_1 and f_2 functions causes that even very small changes in the space of decision variables can cause big differences in the space of objectives. These very features cause that *Kursawe* problem is quite difficult for solving in general—and for solving using evolutionary techniques in particular.

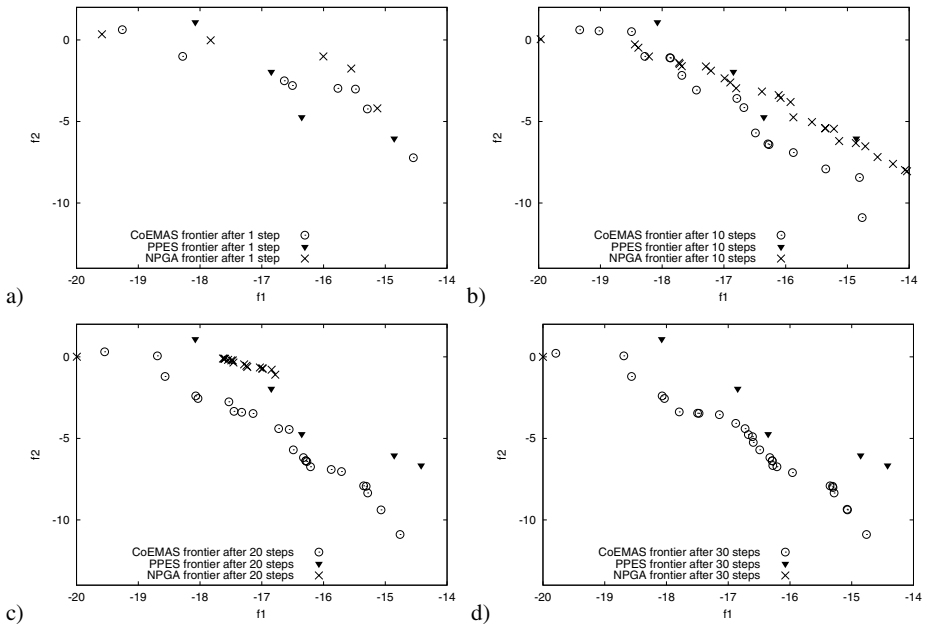


Fig. 2. Pareto frontier approximations obtained by CoEMAS, PPES and NPGA after 1, 10, 20, and 30 steps

For the sake of clarity, “model” Pareto set and frontier will be omitted in further figures presenting their approximations obtained by algorithms that are being analyzed.

As it was mentioned above—Kursawe problem is a quite demanding multi-objective problem with disconnected both Pareto set and Pareto frontier as well. In this section both some qualitative and quantitative characteristics obtained during solving this problem are discussed. To give a kind of reference point, results obtained by CoEMAS are compared with results obtained by “classical” (i.e. non agent-based) predator-prey evolutionary strategy (PPES) [6] and another classical evolutionary algorithm for multi-objective optimization: niched pareto genetic algorithm (NPGA) [10].

In fig. 2 and fig. 3 there are presented approximations of Pareto frontier obtained by all three algorithms that are being analyzed after 1, 10, 20, 30, 50, 100, 600 and 2000 time steps. As one may notice initially, i.e. after 1, 10 and partially after 20 (see fig. 2a, 2b and 2c) steps, Pareto frontiers obtained by all three algorithms are—in fact—quite similar if the number of found non-dominated individuals, their distance to the model Pareto frontier and their dispersing over the whole Pareto frontier are considered. Afterwards yet, definitely higher quality of CoEMAS-based Pareto frontier approximation is more and more distinct. It is enough to mention that NPGA-based Pareto frontier almost completely disappears after about 30 steps, and PPES-based Pareto frontier is—as the matter of fact—better and better but this improving process is quite slow and not so clear as in the case of CoEMAS-based solution.

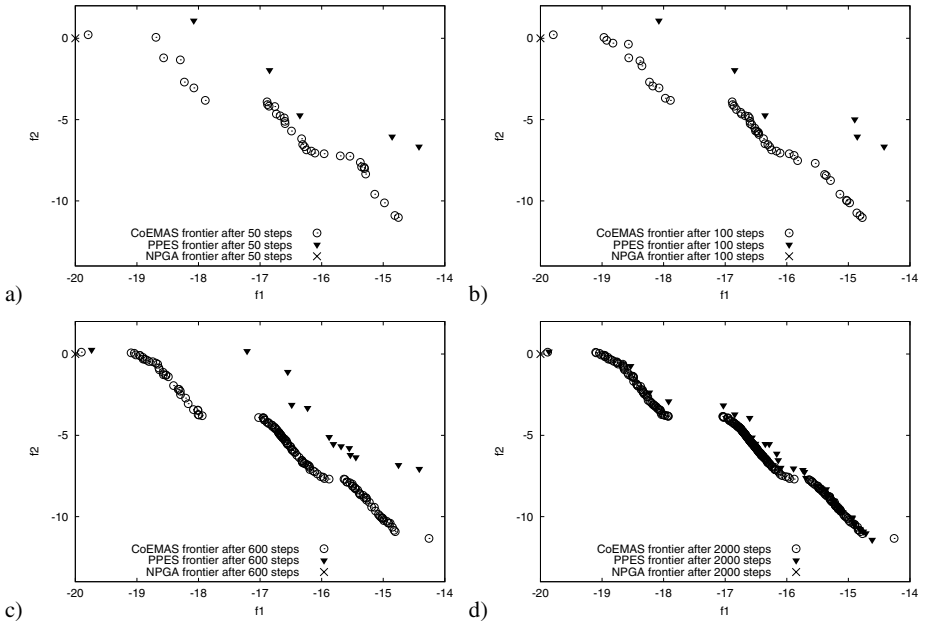


Fig. 3. Pareto frontier approximations obtained by CoEMAS, PPES and NPGA after 50, 100, 600, and 2000 steps

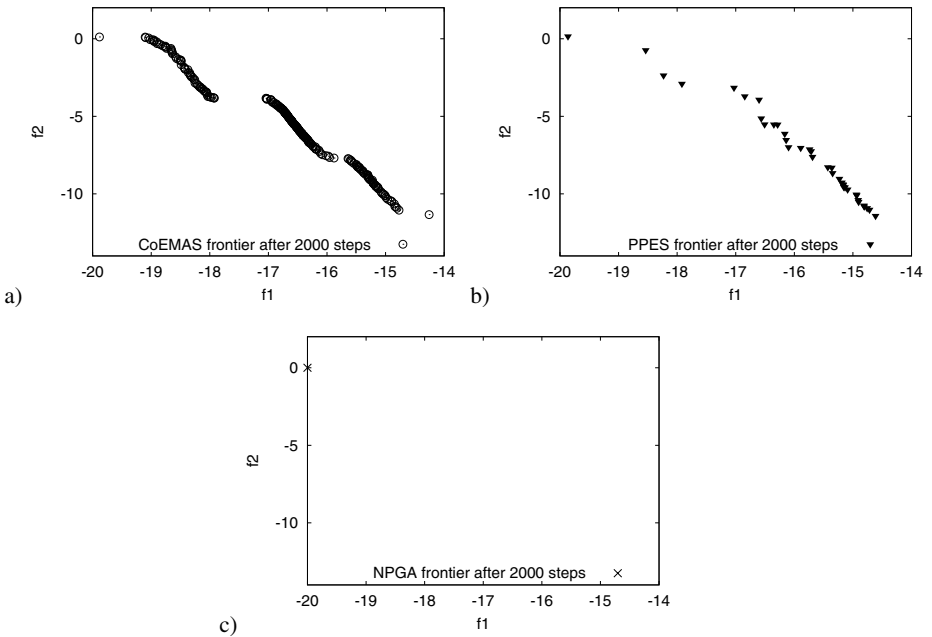


Fig. 4. Pareto frontier approximations obtained by CoEMAS, PPES and NPGA after 2000 steps

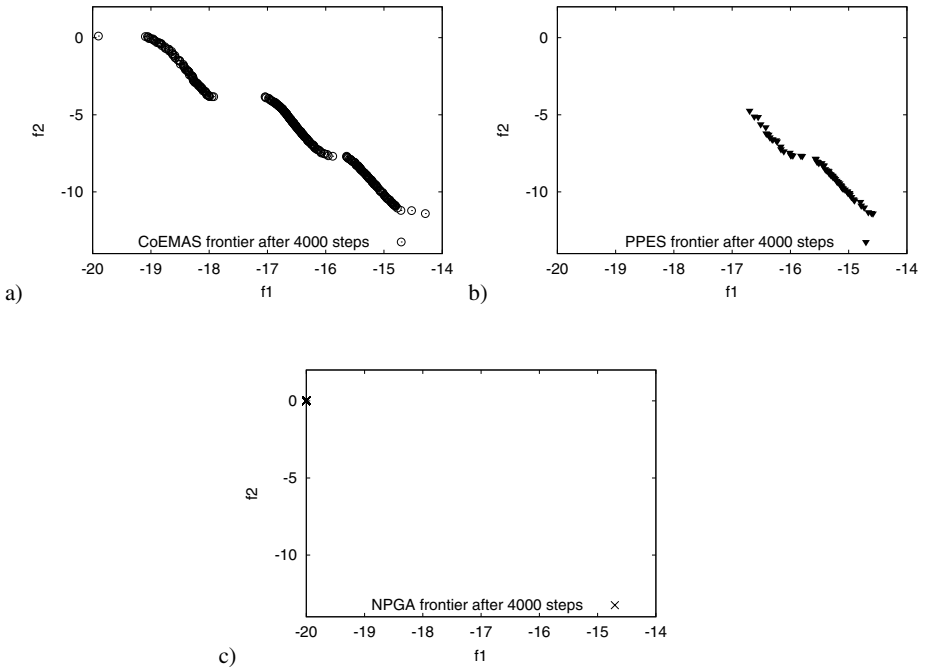


Fig. 5. Pareto frontier approximations obtained by CoEMAS, PPES and NPGA after 4000 steps

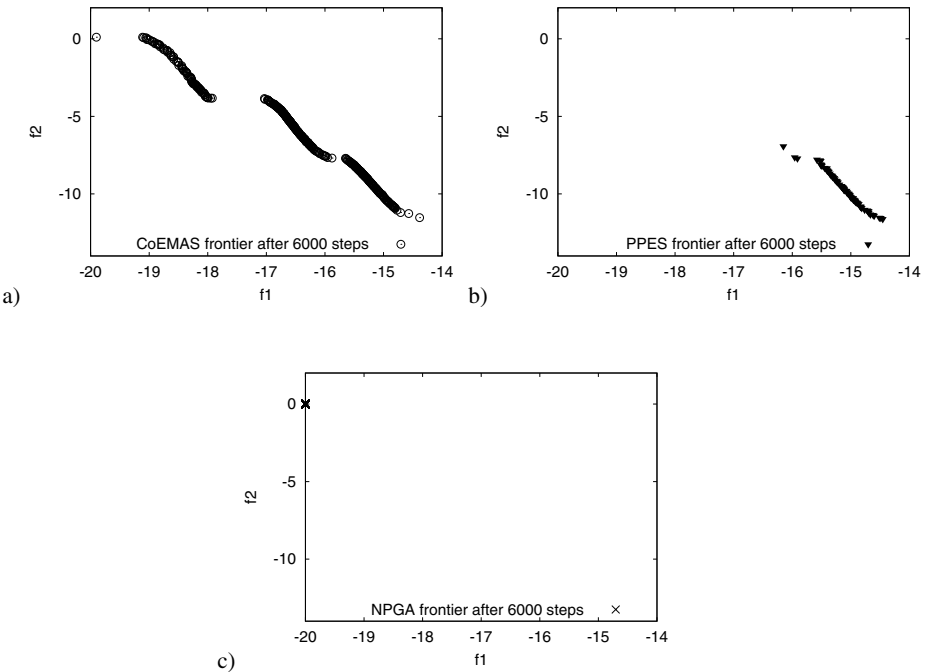


Fig. 6. Pareto frontier approximations obtained by CoEMAS, PPES and NPGA after 6000 steps

Because solutions presented in fig. 2 and in fig. 3 (especially in fig. 3d) partially overlap, in fig. 4, fig. 5 and fig. 6 there are presented Pareto frontiers obtained by analyzed algorithms after 2000, 4000 and 6000 time steps separately. There is no doubt that—what can be especially seen in fig. 4a, fig. 5a, and in fig. 6a—CoEMAS is definitely the best alternative since it is able to obtain Pareto frontier that is located very close to the model solution, that is very well dispersed and what is also very important—it is more numerous than PPES and NPGA-based solutions.

It is of course quite difficult to compare algorithms only on the basis of qualitative results, so in Table 1 there are presented values of HV and HVR metrics (which are described in [2]) measured after 1, 10, 20, 30, 40, 50, 100, 600, 2000, 4000 and 6000 steps. The results presented in this table confirm that in the case of Kursawe problem CoEMAS is much better alternative than “classical” PPES or NPGA.

4 Concluding Remarks

Co-evolutionary techniques for evolutionary algorithms makes it possible to apply such algorithms to solving problems for which it is difficult or even impossible to formulate explicit fitness function. Co-evolutionary techniques are rather rarely used as mechanisms of maintaining useful population diversity or as speciation and niching techniques. However, there has been recently growing interest in co-evolutionary algorithms and in the application of such algorithms to multi-objective optimization problems.

The model of *co-evolutionary multi-agent system* allows co-evolution of several species and sexes. This results in maintaining population diversity and improves adaptive capabilities of systems based on CoEMAS model. In this paper the *co-evolutionary multi-agent system with predator-prey mechanism for multi-objective optimization* has been presented. The system was run against commonly used test problem and compared to classical PPES and NPGA algorithms.

Presented results of experiments with Kursawe problem (as another not presented here results obtained with another mentioned above benchmark problems) clearly show that CoEMAS not only properly located Pareto frontier of this test problem but also the results of this system was better than in the case of two other “classical” algorithms. CoEMAS was able to obtain solutions that were located very close to the “ideal” Pareto frontier, that were very well dispersed and more numerous than PPES and NPGA-based solutions. This was the result of the tendency to maintain high population diversity what could be especially very useful in the case of hard dynamic and multi-modal multi-objective problems (as defined by Deb [2]).

Future work will include more detailed comparison to other classical algorithms with the use of hard multi-modal multi-objective test problems. Also the application of other co-evolutionary mechanisms like symbiosis (co-operative co-evolution) are included in future plans.

References

1. T. Bäck, editor. *Handbook of Evolutionary Computation*. IOP Publishing and Oxford University Press, 1997.
2. K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, 2001.

3. R. Dreżewski. Co-evolutionary multi-agent system with speciation and resource sharing mechanisms. *Computing and Informatics*, 25(4):305–331, 2006.
4. R. Dreżewski and L. Siwik. Co-evolutionary multi-agent system with sexual selection mechanism for multi-objective optimization. In *Proceedings of the IEEE World Congress on Computational Intelligence (WCCI 2006)*. IEEE, 2006.
5. R. Dreżewski and L. Siwik. Multi-objective optimization using co-evolutionary multi-agent system with host-parasite mechanism. In V. N. Alexandrov, G. D. van Albada, P. M. A. Sloot, and J. Dongarra, editors, *Computational Science — ICCS 2006*, volume 3993 of *Lecture Notes in Computer Science*, pages 871–878, Berlin, Heidelberg, 2006. Springer-Verlag.
6. M. Laumanns, G. Rudolph, and H.-P. Schwefel. A spatial predator-prey approach to multi-objective optimization: A preliminary study. In A. E. Eiben, et al., editor, *Parallel Problem Solving from Nature — PPSN V*, volume 1498 of *LNCS*. Springer-Verlag, 1998.
7. J. Paredis. Coevolutionary algorithms. In T. Bäck, editor, *Handbook of Evolutionary Computation, 1st supplement*. IOP Publishing and Oxford University Press, 1998.
8. L. Siwik and M. Kisiel-Dorohinicki. Balancing of production lines: evolutionary, agent-based approach. In *Proceedings of Conference on Management and Control of Production and Logistics*, pages 319–324, 2004.
9. L. Siwik and M. Kisiel-Dorohinicki. Semi-elitist evolutionary multi-agent system for multiobjective optimization. In V. Alexandrov, P. van Albada, G.D. amd Sloot, and J. Dongarra, editors, *Proceedings of the 6th International Conference Computational Science - ICCS 2006, PartIII*, volume 3993 of *LNCS*, pages 831–838. Springer, May 2006.
10. E. Zitzler. *Evolutionary algorithms for multiobjective optimization: methods and applications*. PhD thesis, Swiss Federal Institute of Technology, Zurich, 1999.

Optical Design with Epsilon-Dominated Multi-objective Evolutionary Algorithm

Shaine Joseph, Hyung W. Kang, and Uday K. Chakraborty

Department of Mathematics and Computer Science
University of Missouri, St. Louis

One University Blvd., St. Louis, MO 63121, USA

shaine_joseph@yahoo.com, kang@cs.ums1.edu, uday@cs.ums1.edu

Abstract. Significant improvement over a patented lens design is achieved using multi-objective evolutionary optimization. A comparison of the results obtained from NSGA2 and ϵ -MOEA is done. In our current study, ϵ -MOEA converged to essentially the same Pareto-optimal solutions as the one with NSGA2, but ϵ -MOEA proved to be better in providing reasonably good solutions, comparable to the patented design, with lower number of lens evaluations. ϵ -MOEA is shown to be computationally more efficient and practical than NSGA2 to obtain the required initial insight into the objective function trade-offs while optimizing large and complex optical systems.

1 Introduction

Optical systems consist of a lens system and supporting opto-mechanical instrumentation. They are used in a variety of imaging applications such as telescopes, cameras, projectors, micro-lithography, reconnaissance etc. Lens system design is the most important aspect of optical system design. The quality of imaging is judged by its deviation between the image of an object and the object itself, which can be classified in terms of various ray aberrations. Ray aberrations can be calculated through ray tracing as the distance between the ideal image point and the actual image point on the chosen image plane. The objective of the optimization during lens design is to reduce the aberrations to an acceptable level. An optical system [1] will have a number of constructional parameters (design variables for optimization) to specify the system. These parameters include the radii of curvatures of the lens, the distance between various surfaces etc. Figure 1 illustrates the design variables (curvatures, thicknesses) of a patented Petzval lens design (USA PATENT 2,744,445), whose values are given in Table 1. Petzval lens is very well known, used in the projection of 16mm and 8mm movie films and aerial reconnaissance. Only spherical surfaces are used in the construction of Petzval lens. The apparently simple geometry, wide usage and familiarity are the reasons for selecting Petzval lens system for this study. Also, the results obtained can be easily understood and interpreted.

Lens design is a complex problem since the image quality objective function is a highly non-linear function of the constructional parameters of the system, and the objective function exhibits a large number of local minima. The success of the gradient based, local optimization program depends on the selection of suitable starting

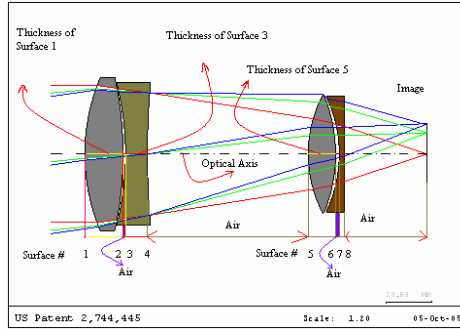


Fig. 1. Petzval Reference Solution from Patent Database is shown, with the following convention. The object is placed on the left side of the system, with the surfaces numbered consecutively from 1 to 8. The radius of curvature, which is the inverse of *curvature*, is positive only if the center of curvature lies to the right of a surface. The *thickness of surface j* is the distance between surface j and $j+1$ measured along the axis of rotational symmetry, called *optical axis*.

Table 1. Surface Data Listing for the reference lens, with the design variables highlighted. The 8-th thickness is not a design variable as it is adjusted (indicated by PIM) to obtain the desired imaging plane. The glass codes, following the optics convention, identify the lens material.

| Surface | Curvature | Thickness | GLASS CODE |
|-----------|-----------|-----------|---------------|
| object: | 0.0000 | INFINITY | Air |
| 1: | 0.0111 | 17.50 | 677410.556000 |
| 2: | -0.0062 | 1.15 | Air |
| 3: | -0.0076 | 10.00 | 688730.312000 |
| 4: (stop) | 0.0027 | 73.80 | Air |
| 5: | 0.0173 | 12.50 | 677410.556000 |
| 6: | -0.0142 | 1.75 | Air |
| 7: | -0.0167 | 2.50 | 741740.275000 |
| 8: | 0.0000 | 37.75 | PIM |
| image: | 0.0000 | 0.00 | |

point. Hence, optimization is often repeated from many different starting points selected on the basis of designer insight. Even if we choose different starting points, we may end up at the same local optimum, and hence may not obtain competing Pareto-optimal designs. Global optimization methods such as genetic algorithms have been applied [2] to avoid the local minima. The recent developments in lens design are automatic addition and deletion of elements during optimization based on power and symmetry distribution [3] among the component elements; genetic programming based automated invention [4], and saddle-point-detection-based approach [5] towards navigating to the global minima. They are all concerned with single objective optimization. However, the image quality expressed as a single number is an approximation for image quality multi-objective functions. Strong domain knowledge and prior knowledge of trade-offs among the objectives are required in accurately formulating this equivalent objective function. Pareto-Optimal solutions in the saddle point regions of equivalent (weighted sum) single-objective optimizations are bound to be undetected [6] in gradient-based optimization. For larger lens systems, the trade-offs available among the objectives in the feasible space are difficult to obtain prior to

optimization. Multi-objective approach, on the other hand, allows the objectives to be loosely articulated. For a chosen layout, optimization results will indicate the trade-offs involved. These factors demand a multi-objective evolutionary approach for large optical system optimization.

In this paper, a Petzval lens is sought to be optimized for image quality in a multi-objective framework using ϵ -Dominated Multi-Objective Evolutionary Algorithm (ϵ -MOEA) [7], [12]. Earlier results on multi-objective evolutionary approach to lens design by Ono [8] lacked diversity on Pareto-Optimal front. Our previous studies [9]-[10] based on hybrid coded NSGA2 [11], [12] have been successful in obtaining a well distributed diverse set of Pareto-optimal solutions that were comparable to or better than the patented design. We had also investigated the Pareto-optima in the saddle point regions of a Petzval lens, and demonstrated that all the positive attributes of the previously mentioned single objective approaches can be integrated in a multi-objective framework [9] to yield enhanced functionality. However convergence of NSGA2 is slow. Multi-objective evolutionary approach, when applied to optimize large optical systems, needs faster convergence to be effective. Theoretical studies using test problems [7] have indicated that ϵ -MOEA converges to Pareto-optima quicker than NSGA2 at the expense of diversity and uniformity of distribution. However, here we have a dominant objective function and two ancillary objective functions [9], [10]. This renders the diversity and uniformity of distribution in objective space subordinate to achieving convergence to Pareto-optimal front with least amount of computation. The reasons for selecting Petzval lens system for this study are its wide usage and familiarity that enable clear interpretation of the results. The results are compared against that obtained for NSGA2 and known patented (Fig.1) designs.

1.1 Problem Definition

The objective of this optimization is to obtain Pareto-optimal solutions for a Petzval lens with effective focal length (EFL) of 100 mm and F# (F-number) of 1.5953 over an object field angle of 8 degrees. The wavelengths under consideration are 656.3 nm, 587.6 nm, and 486.1 nm where 587.6 nm is used as the reference wavelength. The optical design program CODE V[®] is employed as a ray tracer that calculates the objective functions. A Petzval lens is sought to be optimized for its image quality. The Petzval lens from CODE V[®] patent database (USA PATENT 2,744,445) was employed as a reference solution (see Fig.1). The *design variables* are curvatures and thicknesses. We have 8 radii of curvatures, in the domain [-0.03 /mm, 0.03 /mm]. The fourth thickness varies in [0 mm, 200 mm] whereas the remaining 6 thicknesses vary in the domain [0 mm, 20 mm]. All variables are real coded. However, the last thickness is adjusted such that the image is formed on Paraxial Image Plane and hence it is not a variable for optimization. The three objectives (Sect. 1.2) to be minimized are absolute values of spherical aberration and distortion and the CODE V[®] default objective function, which is the dominant objective.

1.2 Objective Functions Definition

Let the system have M aberrations defined over a space of N constructional parameters. The objective function Φ for single objective optimization is usually defined by:

$$\Phi(X) = \sum_{i=1}^M f_i^2(X), \text{ where } X = (x_1, x_2, \dots, x_N), \quad (1)$$

where f_i are the aberrations and X is a vector of constructional parameters. With multi-objective optimization, the merit function shown above is split into component objective functions. In our study, the objective functions to be minimized are:

$$f_1 = (SA), \quad f_2 = (DST), \quad \text{and} \quad f_3 = (ERR), \quad (2)$$

where (SA) and (DST) stand for Spherical Aberration and Distortion. Here (ERR) is the CODE V[®] default objective function computed over a large number of rays (12 rays from each field). This is a nonlinear weighted sum of aberrations of many rays over different fields and wavelengths. Construction of the default objective function takes into account every type of aberrations and provides the most reliable single indicator of image quality. Spherical aberration depends only on the curvatures of lens surfaces whereas distortion is maximally dependent on the field angle of incident light and least dependent on the curvature. This particular dependence of chosen ancillary objective functions on design variables helps us maintain a diverse set of relevant phenotypes and genotypes in a multi-objective framework. In almost all useful optical systems, minimization of this dominant objective function results in the minimization of the ancillary objective functions as well.

2 Methodology

There are various multi-objective algorithms available. In all of them, there is a trade-off [7] between achieving a well-converged, and well-distributed Pareto-optimal solutions and reducing the computation. Here we apply NSGA2 and ϵ -MOEA [7], both developed by Deb, to optimize a Petzval lens. Both algorithms proceed through cycles of *selection*, *cross-over*, *mutation*, and *objective function evaluation* over generations. ϵ -MOEA achieves a compromise among the above objectives as to obtain reasonably well-distributed Pareto-optimal solutions quickly. Here we have two co-evolving populations. One is an archive of non-dominated solutions, with a varying size, which gets updated after every generation according to the ϵ -domination concept. The concept of ϵ -non-domination means that no two Pareto-optimal solutions are allowed which differ by less than ϵ_i for the i^{th} objective. The other is a population of fixed size, which gets updated every generation according to the usual domination concept. The selection process for cross-over is such that one parent is chosen from the archive at random. The other parent is selected from the co-evolving population based on a binary-tournament between two randomly selected solutions. This is a steady state, diversity preserving and elitist multi-objective algorithm, which additionally allows us to choose the desired resolutions for every objective. NSGA2, on the other hand, provides well-converged and well-distributed Pareto-optimal solutions at a greater computational cost. NSGA2 maintains a population of fixed size throughout generations. The selection mechanism explicitly maintains *elitism* and *diversity* among parents, achieved through *non-dominated* and *crowded distance sorting*.

ϵ -MOEA maintains a population size of 200. Every generation evaluates 2 offspring solutions. For the random creation of the initial population and the first 500 generations, we allow only scaling. For the next 500 generations, we allow 1 step of local optimization for the dominant objective function. For the subsequent generations, we allow 2 steps of local optimization. NSGA2 needs to have a very large population size to start with, during the random initialization. Otherwise, we may not obtain diverse initial feasible solutions. For example, when 1000 solutions were randomly created and subsequently, another 1000 was created through cross-over and mutation, only 5 solutions were feasible after the first generation, whereas the initial random creation of 600 solutions in a separate trial gave only 1 solution after the first generation.

The cross-over probability was selected as 1 whereas the mutation probability was 1/15. The distribution index for cross-over and mutations were chosen as 15 and 20 respectively for ϵ -MOEA. The chosen resolution $\epsilon = [0.01, 0.01, 10]$. The cross-over distribution index was selected as 200 for NSGA2. The source code from Deb [12] was used with the required modifications.

Whenever a new solution is generated, we *scale* [10] the lens design variables to achieve an EFL of exactly 100mm. The new scaled solution can be accepted only if it falls within the designated domain of design variables (Sect. 1.1). However, scaling often takes a solution outside of the designated domain. It also worsens the aberrations [10]. Hence, following scaling, limited steps of local, gradient based, single objective optimization for the dominant objective function is done subject to all the constraints mentioned above, except the curvatures. If the resultant solution falls within the accepted domain for curvatures, the new solution is accepted for checking Pareto-optimality. Otherwise, the original solution before scaling is retained, with all objective vectors and constraint violations assigned a very high value, representing the nadir objective vector.

3 Results

NSGA2 requires a very large population size during the random initialization. Otherwise, we do not get sufficient number of diverse feasible solutions to start the evolutionary optimization. This is illustrated while comparing fit 2 and 3 in (Fig.2). ϵ -MOEA results are shown in fit 1. Evolution starting at 500th generation (1200th solution evaluation) is shown. Fit 2 and fit 3 show the NSGA2 starting at 1400th and 2200th solution evaluations respectively. The following observations can be made.

With the evaluation of 1200 solutions in the first generation of NSGA2, we obtained only 1 feasible solution. For reducing the computational effort, NSGA2 was restarted with this 1 feasible solution obtained above as seed, apart from another 19 infeasible dummy solutions making up a population size of 20. The minima obtained had a value of 3583 (Fig. 5(b)) after 3500 evaluations, as opposed to 3095, obtained for the patented design. The evolution is shown in fit 2 of Fig. 2. Between 3600-4200 evaluations, the minima reached and remained at 3454 indicating convergence

With the evaluation of 2000 solutions in the first generation of NSGA2, we obtained 5 distinct feasible solutions. In order to reduce the computational effort, NSGA2 was restarted with the 5 feasible solutions obtained above as seed, apart from

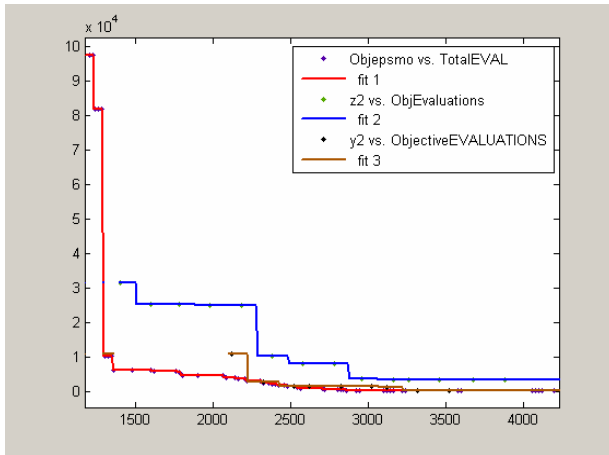


Fig. 2. The dominant objective function (*ERR*) as it evolves over generations for 1 trial of ϵ -MOEA and 2 trials of NSGA2 are shown

another 15 infeasible dummy solutions making up a population size of 20. The final solutions obtained have beaten the patented solution and gave the value of the dominant objective function as 395 as opposed to 3095, obtained for the patented design. The evolution is shown in fit 3 of Fig.2.

ϵ -MOEA converges to the optima faster. The population required is moderate. It is 200 as opposed to 1000, initially required for NSGA2. The final solutions obtained have beaten the patented solution, and gave the dominant objective function as 384 (Fig. 5(a)) after 3500 evaluations, as opposed to 3095, obtained for the patented designs. The evolution is shown in fit 1 of Fig. 2. Another 700 evaluations (totaling 4200) gave a marginal improvement in objective function, at 379, indicating convergence.

Figures 3-5 show the solutions at 3 distinct ranges of lens evaluations (1400, 2100-2200, 3500-3520) of Fig.2. When we do not have any restrictions on the computational resources NSGA2 and ϵ -MOEA performed equally well for the given problem. However, ϵ -MOEA is better suited to provide reasonably good solutions, comparable to the patented reference solution, with lower number of lens evaluations.

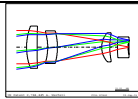
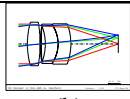
| | ϵ -MOEA Evaluation #1400 | NASG2 (fit2) Evaluation #1400 |
|------------|--|--|
| |  (a) |  (b) |
| <i>SA</i> | 0.056 | 0.385 |
| <i>DST</i> | 0.720 | 0.026 |
| <i>ERR</i> | 6241 | 25419 |

Fig. 3. Comparison of algorithms after 1400 lens evaluations

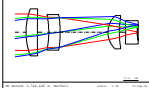
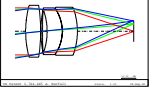
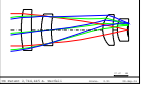
| | ϵ -MOEA Evaluation #2120 | NASG2 (fit2) Evaluation #2180 | NASG2 (fit3) Evaluation #2120 |
|------------|---|---|---|
| |  |  |  |
| | (a) | (b) | (c) |
| <i>SA</i> | 0.011 | 0.436 | 0.124 |
| <i>DST</i> | 1.071 | 0.027 | 0.620 |
| <i>ERR</i> | 4066 | 25049 | 11038 |

Fig. 4. Comparison of algorithms after 2100-2200 lens evaluations

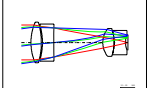
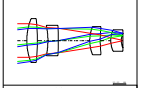
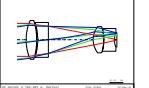
| | ϵ -MOEA Evaluation #3500 | NASG2 (fit2) Evaluation #3500 | NASG2 (fit3) Evaluation #3520 |
|------------|---|---|---|
| |  |  |  |
| | (a) | (b) | (c) |
| <i>SA</i> | 0.107 | 0.021 | 0.106 |
| <i>DST</i> | 0.672 | 0.440 | 0.663 |
| <i>ERR</i> | 384 | 3583 | 395 |

Fig. 5. Comparison of algorithms after 3500-3520 lens evaluations

4 Conclusions

A Petzval lens is optimized for image quality in a classical-evolutionary hybrid framework. The results obtained from NSGA2 and ϵ -MOEA is compared. Typically, ϵ -MOEA converges to Pareto-optima quicker than NSGA2 at the expense of diversity and uniformity of distribution. However, for the problem at hand, we have a dominant objective function and two ancillary objective functions. The particular dependence of chosen ancillary objective functions on *design variables* helps us maintain a diverse set of relevant phenotypes and genotypes in a multi-objective frame work. In almost all useful optical systems, minimization of this dominant objective function results in the minimization of the chosen ancillary objective functions as well. In our current study, ϵ -MOEA converged to essentially the same Pareto-optimal solutions as the one with NSGA2, but ϵ -MOEA proved better to provide reasonably good solutions, comparable to the patented design, with lower number of lens evaluations. ϵ -MOEA has an added advantage of specifying desired resolution of the Pareto-optimal front for the individual objectives. ϵ -MOEA is better suited than NSGA2 to obtain the required initial insight into the objective function trade-offs while optimizing large and complex optical systems.

Acknowledgements. This research was partially supported UM Research Board and the MIC (Ministry of Information and Communication), Korea, under ITRC (Information Technology Research Center) support program supervised by IITA (Institute of Information Technology Assessment). S. Joseph thanks the Optical Research Associates for granting a student license of CODE V®.

References

1. Kidger, M.J.: *Intermediate Optical Design*, SPIE Press (2004)
2. Vasiljevic, D.: *Classical and Evolutionary Algorithms in the Optimization of Optical Systems*. Kluwer Academic Publishers (2002)
3. Cheng, X., Wang, Y., Hao, Q., Sasian, J.: Automatic element addition and deletion in lens optimization. *Applied Optics*. 42(7) (2003) 1309-1317
4. Koza, J.R., Al-Sakran, S.H., Jones, L.W.: Automated re-invention of six patented optical lens systems using genetic programming. *Proc. GECCO*, New York (2005) 1953–1960
5. Bociort, F., Driel, E., Serebriakov, A.: Network structure of the set of local minima in optical system optimization. *Proc. of SPIE*, Vol. 5174 (2003) 26-34
6. Deb, K.: *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons Ltd (2003)
7. Deb, K., Mohan, M., Mishra, S.: A Fast Multi-objective Evolutionary Algorithm for Finding Well-Spread Pareto-Optimal Solutions. *KanGAL Report No. 2003002* (2003)
8. Ono, S., Kobayashi, Yoshida, K.: Global and Multi-objective Optimization for Lens Design by Real-coded Genetic Algorithms. *Proc. of SPIE*, Vol. 3482 (1998) 110-121
9. Joseph, S.: *Lens Design and Optimization Using Multi-Objective Evolutionary Algorithms*. Ph.D. dissertation, University of Missouri-Rolla (2005)
10. Joseph, S., Kang, H.W., Chakraborty, U.K.: Lens Optimization In A Classical-Evolutionary Hybrid Framework. *Proc. MENDEL: 12th International Conference on Soft Computing*, Brno, Czech Republic, EU (2006)
11. Coello, C.A.C., et al.: *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers (2002)
12. <http://www.iitk.ac.in/kangal/soft.htm>

Boosting the Performance of a Multiobjective Algorithm to Design RBFNNs Through Parallelization

Alberto Guillén¹, Ignacio Rojas², Jesus González², Hector Pomares²,
Luis J. Herrera², and Ben Paechter³

¹ Department of Computer Science
University of Jaén, Spain
aguillen@atc.ugr.es

² Department of Computer Architecture and Computer Technology
Universidad de Granada, Spain

³ School of Computing
Napier University, Scotland

Abstract. Radial Basis Function Neural Networks (RBFNNs) have been widely used to solve classification and regression tasks providing satisfactory results. The main issue when working with RBFNNs is how to design them because this task requires the optimization of several parameters such as the number of RBFs, the position of their centers, and their radii. The problem of setting all the previous values presents many local minima so Evolutionary Algorithms (EAs) are a common solution because of their capability of finding global minima. Two of the most important elements in an EAs are the crossover and the mutation operators. This paper presents a comparison between a non distributed multiobjective algorithm against several parallel approaches that are obtained by the specialisation of the crossover and mutation operators in different islands. The results show how the creation of specialised islands that use different combinations of crossover and mutation operators could lead to a better performance of the algorithm by obtaining better solutions.

1 Introduction

Designing an RBFNN to approximate a function from a set of input-output data pairs, is a common solution since this kind of networks are able to approximate any function [1]. Formally, a function approximation problem can be formulated as, given a set of observations $\{(\mathbf{x}_k; y_k); k = 1, \dots, n\}$ with $y_k = F(\mathbf{x}_k) \in \mathbb{R}$ and $\mathbf{x}_k \in \mathbb{R}^d$, it is desired to obtain a function \mathcal{G} so $y_k = \mathcal{G}(\mathbf{x}_k) \in \mathbb{R}$ with $\mathbf{x}_k \in \mathbb{R}^d$. Once this function is learned, it will be possible to generate new outputs from input data that were not specified in the original data set.

A RBFNN \mathcal{F} with fixed structure to approximate an unknown function F with n entries and one output starting from a set of values $\{(\mathbf{x}_k; y_k); k = 1, \dots, n\}$ with $y_k = F(\mathbf{x}_k) \in \mathbb{R}$ and $\mathbf{x}_k \in \mathbb{R}^d$, has a set of parameters that have to be optimized:

$$\mathcal{F}(\mathbf{x}_k; C, R, \Omega) = \sum_{j=1}^m \phi(\mathbf{x}_k; \mathbf{c}_j, r_j) \cdot \Omega_j \quad (1)$$

where m is the number of RBFs, $C = \{\mathbf{c}_1, \dots, \mathbf{c}_m\}$ is the set of RBF centers, $R = \{r_1, \dots, r_m\}$ is the set of values for each RBF radius, $\Omega = \{\Omega_1, \dots, \Omega_m\}$ is the set of weights and $\phi(\mathbf{x}_k; \mathbf{c}_j, r_j)$ represents an RBF. The activation function most commonly used for classification and regression problems is the Gaussian function because it is continuous, differentiable, it provides a softer output and improves the interpolation capabilities [2]. The procedure to design an RBFNN for functional approximation problem is: 1) Set the number of RBFs in the hidden layer 2) Place RBF centers \mathbf{c}_j 3) Initialize the radius r_j for each RBF 4) Calculate the optimum value for the weights Ω_j .

We want to find both a network with the smallest number of neurons and one with the smallest error. This is a multiobjective optimisation problem because for some pairs of networks it is impossible to say which is better (one is better on one objective, one on the other) making the set of possible solutions partially sorted.

2 Multiobjective Algorithm for Function Approximation: MOFA

One of the most popular multiobjective algorithms is the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) [3] since it presents a good performance in several applications [4]. Therefore, an adaptation of this algorithm to design RBFNNs is presented in this section.

2.1 Initial Population

The individuals in the population of the algorithm will contain the position of the centers and the length of the radii in a vector of real numbers. The storage of the weights in the chromosome is useless since they can be obtained optimally by solving a linear equation system, and for each change in a radius or a center they have to be updated. In order to start from an adequate position in the large solution space, the individuals are initialized using clustering algorithms [5] [6] to set the positions of the centers and then, the length of the radii is chosen randomly. The initial population also includes individuals generated randomly in order keep diversity in the population.

Once half of the population has been initialized, a few iterations of a local search algorithm [7] are applied to each individual adjusting its centers and radii, then the results are appended to the population. This procedure increments the diversity and, as experimentally has been proven, improves the quality of the results.

The initial size of the population should be small (not more than 5 RBFs) because we want the clustering algorithms and the local search algorithm to

execute as fast as possible and, since the algorithm also optimizes the size of the networks, we save computational time if we avoid dealing with big networks in the first generations.

2.2 Crossover Operators

The original crossover operator over a binary or real coded chromosome cannot be performed with the individuals of the proposed algorithm because each gene represents different elements. Thus, specific crossover operators have been designed considering complete RBFs as genes to be exchanged by the chromosomes representing RBFNNs.

Crossover operator 1: Neurons exchange. This crossover operator, conceptually, would be the most similar one to a standard crossover because the individuals represent an RBFNN with several neurons and a crossover between two individuals would result in a neuron exchange. The operator will exchange only one neuron, selected randomly, between the two individuals.

This crossover operator exploits the genetic material of each individual in a simple and efficient way without modifying the structure of the network.

Crossover operator 2: Addition of the neuron with the smallest local error. This operator consists in the addition into one parent of the neuron with the smallest local error belonging to the other parent. The local error of a neuron is defined as the sum of the errors between the real output and the output generated by the RBFNN for the input vectors that activate that neuron.

To make sure that the offsprings are different of their ancestors, before adding the neuron with the smallest local error, the crossover will make sure that the neuron to be added does not exist in the individual, otherwise the second neuron with the smallest local error will be considered and so on. This restriction combined with the way the NSGA-II proceeds helps to avoid the “competing convention” problem [8].

Once the offspring are obtained, they go through a refinement process that consists in the prune of the RBFs which doesn’t influence the output of the RBFNN, to do this, all the weights that connect the processing units to the output layer are calculated and the neurons that don’t have a significant weight will be removed.

2.3 Mutation Operators

The mutation operators add randomness into the process of finding good solutions. These mutations allow the algorithm to explore the solution space avoiding being trapped in local minima. The modifications can be purely random or can be performed using problem specific knowledge. The mutation operators are desired to be as much simple as they can so it can be shown clearly, without the interference of introducing expert knowledge, the effects of the parallelization at a very basic level. For an RBFNN, two kind of modifications can be performed:

- Changes in the structure of the RBFNN (Increase and decrease operators): Addition and Deletion of an RBF. The first one adds an RBF in one random position over the input vectors space, setting its radius with a random value. All the random values are taken from an uniform distribution in the interval $[0,1]$ since the input vectors and their output are normalized. The second operator is the opposite to the previous one, deleting a random existing RBF from the network. This mutation is constrained so that is not applied when the individual has less than two neurons.
- Changes in the elements of the structure (Movement operators): The third and the fourth operators refer to real coded genetic algorithms as presented in [9]. The third operator adds to all the coordinates of a center a random distance chosen in the interval $[-0.5,0.5]$ from a uniform distribution. The fourth one has exactly the same behavior but changing the value of the radius of the selected RBF.

3 Island Specialisation

In order to obtain results of maximum quality and take advantage of each type of operator, several parallel implementations were studied. All these implementations are based on the island paradigm where there are several islands (in our parallel implementation these map to processors) that evolve independent populations and, exchange information or individuals.

The design of an RBFNN, involves two well defined tasks: the design of the structure and the determination of the value of the parameters that build that structure. From this point of view many island specialisations were analysed although in this paper only the more representative ones are commented:

1) *Division of the crossover operators (P1)*. This approach allows specialisation of the first stage of the evolution process, that is, the reproduction process. The two crossover operators are separated and executed in different islands. The mutation process remains the same for both types of islands. The following algorithms study the possible ways of specialising the mutation stage but always using the specialised crossover topology. As it will be shown in the experiments, allowing specialisation of the crossover operators gives significant improvements.

2) *Crossover 1 + Movement + Decreasing and Crossover 2 + Movement + Increasing (P2)*. This combination of operators aims to boost the exploration capabilities of the crossover 2 and the exploitation ones of crossover 1. Since the crossover 2 explore more topologies by increasing the size of the NNs, the specialisation can be done by letting this island to produce only bigger networks. The other island with the crossover 1, will take care of exploitation of the chromosomes and will decrease the size of the networks.

3) *Crossover 1 + Movement + Increasing and Crossover 2 + Movement + Decreasing (P3)*. This algorithm is the opposite to the previous one. The aim is to not allow the crossover 2 to create too big NNs by adding the operator that removes an RBF. If the NNs in the population become too big, the computational time would be highly increased. To be able to explore more different topologies,

the island with the crossover 1 will increase the number of RBFs using the increasing mutation operator.

4) *Crossover 1 + Movement and Crossover 2 + Increasing and Decreasing (P4)*. This approach is the one that specialises the most each island on each task of the design of an RBFNN. The island with the crossover 1 will take care of changing the parameters of the centers and the radii and at the same time, will exploit the genetic information contained in one topology through its crossover operator. The second island will just take care of the modifications on the structure of the RBFNN by increasing or decreasing its size through both the crossover and mutation operators.

4 Experiments

This section analyzes the behavior of the algorithms described above to show that specialisation of islands to use different operators could lead to better results. The experiments were performed using a two dimensional function that was generated using a Gaussian RBFNN with randomly chosen parameters ($\exp(-\frac{\|\mathbf{x}_k - \mathbf{c}_i\|^2}{r_i^2})$) over a grid of 25x25 points.

Although some metrics have been applied to compare nondominated sets [10], in [11] it is suggested that: “as long as a derived/known Pareto front can be visualized, pMOEA¹ effectiveness may still best be initially observed and estimated by the human eye”. Therefore the experimental analysis of the algorithms will be based on the plots of the resulting Pareto fronts.

The algorithms were executed using the same initial population of the same size and the same values for all the parameters. The crossover probability was 0.5 and the mutation probability was 0.25, the size of the population 100 and the number of generations was fixed to 100. The probabilities might seem high but it’s necessary to have a high mutation probability to allow the network to increase its size, and the centers and radii to move. The high crossover probability makes it possible to show the effects of the crossover operators. Several executions were made changing the migration rate, allowing the algorithms to have 2 (each 40 generations), 4 (each 20 generations), 9 (each 10 generations), and 19 (each 5 generations) migrations through the 100 generations.

4.1 Parallel Approach P1 vs. Non Distributed Approaches

The non distributed algorithms are considered as those that have no communication. The three different algorithms are determined by the crossover operators and all of these algorithms will use the four mutation operators at the same time. The first algorithm uses only the crossover 1, the second algorithm uses the crossover 2 and the third non distributed algorithm chooses randomly between the two crossover operators each time it has to be applied. Figure 1 shows that the parallel model outperforms the three previous non distributed approaches.

¹ pMOEA stands for parallel MultiObjective Evolutionary Algorithms.

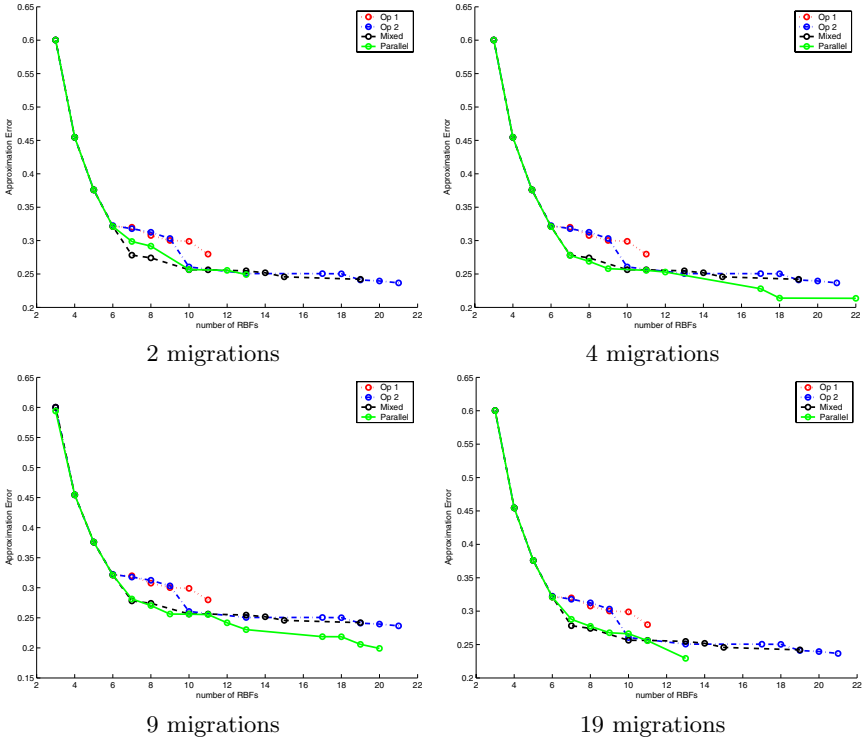


Fig. 1. Pareto front obtained performing several migration steps for P1 and the non distributed approaches

4.2 Comparison Between Parallel Approaches (P1, P2, P3, and P4)

Once it was shown that the parallel approach could lead to better solutions, the different parallel algorithms were compared. Figure 2 shows the results of the four parallel algorithms studied using different migration rates.

The results show how the parallel implementation P3 is not a good choice due to its bad performance. This implementation is the less specialised from all the other ones because the second type of island that uses the crossover 2 is able to reduce the number of RBFs, annulling the effect of the crossover 2 that increases the size of the RBFNNs. For the first island it only increases the size of the network so it won't be able to exploit deeply the genetic recombination.

The algorithms P2 and P4 could be considered the most specialised because their mutation operators are oriented to only one task (P4) or to complement the features of the crossover operators on each island, not like in P3 where the mutation operators counteract the effects of the crossovers. For P2, the best results are obtained for few migrations not like with P4 that achieves the best results with the maximum number of migrations. If the best results for both algorithms are compared, P2 obtains slightly better results for smaller networks

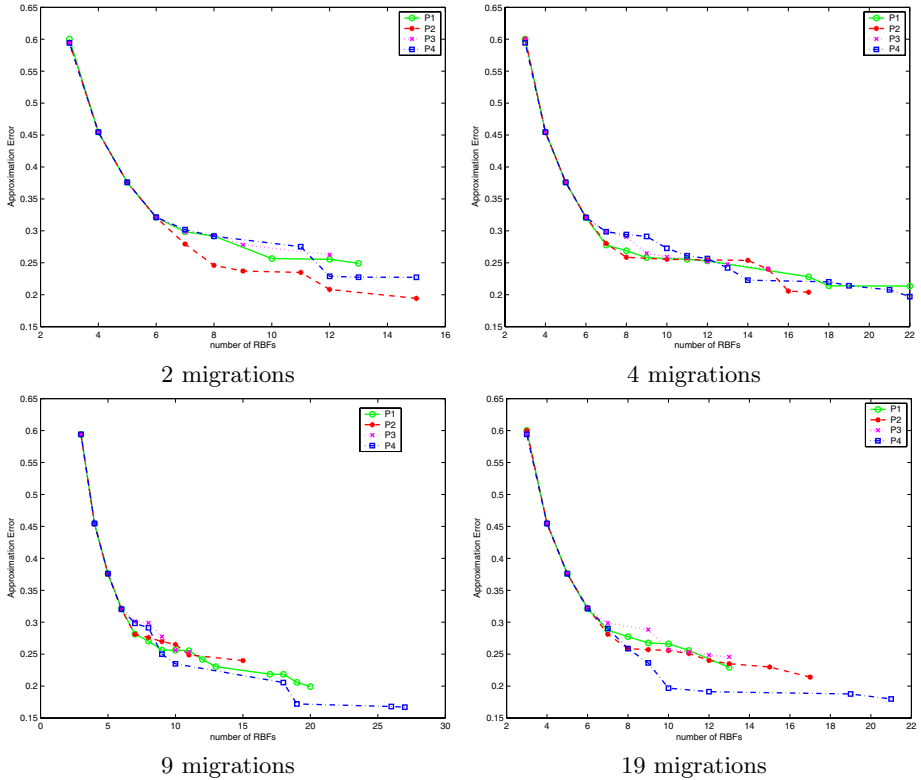


Fig. 2. Pareto front obtained performing several migration steps for P1, P2, P3 and P4

and P4 obtains better results for larger networks and it finds more elements in the Pareto. The behavior of these two algorithms is logical, P2 is able to exploit more the solutions since it reduces the size of the networks and recombines the genes in a more exhaustive way, so it achieves the best results when there are not many migration steps. P4 explores more topologies than P2, that is why it obtains a more complete Pareto, the reason of why it gets the best results with the highest migration rate is because the exploration island must have a fast feedback about the exploitation of the individuals in its Pareto, although there is a limit when if the migration rate becomes too frequent, the islands don't have the chance to exploit and explore, decreasing the quality of the results.

5 Conclusions

Evolutionary approaches are able to solve successfully the complex task of the design of RBFNNs. In this framework, several implementations of a multiobjective genetic algorithm were presented. These approaches emerge as a straight forward specialisation of the crossover and mutation operators in different islands. These

specialisations have the purpose of dividing the exploration and the exploitation tasks between the islands, that share their individuals by migration mechanism. The sharing of the individuals allows the algorithm to keep the characteristic of global optimiser. The experiments showed that, if the different islands evolve specific aspects of the RBFNNs, the results could be improved.

Acknowledgments. This work has been partially supported by the Spanish CICYT Project TIN2004-01419.

References

1. Park, J., Sandberg, J.W.: Universal approximation using radial basis functions network. *Neural Computation* **3** (1991) 246–257
2. Rojas, I., Anguita, M., Prieto, A., Valenzuela, O.: Analysis of the operators involved in the definition of the implication functions and in the fuzzy inference process. *International Journal of Approximate Reasoning* **19** (1998) 367–389
3. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evolutionary Computation* **6**(2) (2002) 182–197
4. Tang, Y., Reed, P., Wagene, T.: How effective and efficient are multiobjective evolutionary algorithms at hydrologic model calibration? *Hydrology and Earth System Sciences* **10** (2006) 289–307
5. Guillén, A., Rojas, I., González, J., Pomares, H., Herrera, L., Valenzuela, O., Prieto, A.: A Possibilistic Approach to RBFN Centers Initialization. *Lecture Notes in Computer Science* **3642** (2005) 174–183
6. Guillén, A., Rojas, I., González, J., Pomares, H., Herrera, L., Valenzuela, O., Prieto, A.: Improving Clustering Technique for Functional Approximation Problem Using Fuzzy Logic: ICFA algorithm. *Lecture Notes in Computer Science* **3512** (2005) 272–280
7. Marquardt, D.W.: An Algorithm for Least-Squares Estimation of Nonlinear Inequalities. *SIAM J. Appl. Math.* **11** (1963) 431–441
8. Hancock, P.J.B.: Genetic Algorithms and Permutation Problems: a Comparison of Recombination Operators for Neural Net Structure Specification. In Whitley, D., ed.: *Proceedings of COGANN workshop, IJCNN, Baltimore, IEEE* (1992)
9. Herrera, F., Lozano, M., Verdegay, J.L.: Tackling real-coded genetic algorithms: operators and tools for the behavioural analysis. *Artificial Intelligence Reviews* **12**(4) (1998) 265–319
10. Knowles, J., Corne, D.: On metrics for comparing non-dominated sets (2002) In *Congress on Evolutionary Computation (CEC 2002)*.
11. van Veldhuizen, D.A., Zydallis, J.B., Lamont, G.B.: Considerations in engineering parallel multiobjective evolutionary algorithms. *IEEE Trans. Evolutionary Computation* **7**(2) (2003) 144–173

Immune Algorithm Versus Differential Evolution: A Comparative Case Study Using High Dimensional Function Optimization

Vincenzo Cutello¹, Natalio Krasnogor², Giuseppe Nicosia¹, and Mario Pavone¹

¹ Department of Mathematics and Computer Science
University of Catania

Viale A. Doria 6, 95125 Catania, Italy

{vctl,nicosia,mpavone}@dmi.unict.it

² Automated Scheduling, Optimisation and Planning Research Group

School of Computer Sciences and IT

Jubilee Campus, University of Nottingham

Nottingham, NG81BB, UK

Natalio.Krasnogor@nottingham.ac.uk

Abstract. In this paper we propose an immune algorithm (IA) to solve high dimensional global optimization problems. To evaluate the effectiveness and quality of the IA we performed a large set of unconstrained numerical optimisation experiments, which is a crucial component of many real-world problem-solving settings. We extensively compare the IA against several Differential Evolution (DE) algorithms as these have been shown to perform better than many other Evolutionary Algorithms on similar problems. The DE algorithms were implemented using a range of recombination and mutation operators combinations. The algorithms were tested on 13 well known benchmark problems. Our results show that the proposed IA is effective, in terms of accuracy, and capable of solving large-scale instances of our benchmarks. We also show that the IA is comparable, and often outperforms, all the DE variants, including two Memetic algorithms.

1 Introduction

Since in many real-world engineering and technology applications analytical solutions, even for simple problems, are not always available, numerical continuous optimisation is often the only viable alternative.

A global minimization problem can be formalized as a pair (S, f) , where $S \subseteq \mathbb{R}^n$ is a bounded set on \mathbb{R}^n and $f : S \rightarrow \mathbb{R}$ is an n -dimensional real-valued function. The goal is to find a point $x_{\min} \in S$ such that $f(x_{\min})$ is a global minimum on S , i.e. $\forall x \in S : f(x_{\min}) \leq f(x)$.

The problem of continuous optimisation is a difficult one not least because it is difficult to decide when a global (or local) optimum has been reached but also because there could be very many local optima that traps the search algorithm. Furthermore, as the dimensionality of the problem increases the number of local optima grows dramatically.

In this work we consider the following numerical minimization problem: $\min(f(\mathbf{x}))$, $\mathbf{B}_l \leq \mathbf{x} \leq \mathbf{B}_u$ where $\mathbf{x} = (x_1, x_2, \dots, x_n)$ is the variable vector in \mathbb{R}^n , $f(\mathbf{x})$ denotes the objective function to *minimize* and $\mathbf{B}_l = (B_{l_1}, B_{l_2}, \dots, B_{l_n})$, $\mathbf{B}_u = (B_{u_1}, B_{u_2}, \dots, B_{u_n})$ represent, respectively, the variables' lower and the upper bounds, such that $x_i \in [B_{l_i}, B_{u_i}]$.

We use the above formulation to evaluate our immune algorithm (IA), first proposed in [1], for high dimensional problems. Moreover, we compare the results of the IA with several *Differential Evolution* (DE) variants as those proposed in [6], [4] and [5]. DE were chosen as they typically show better convergence behaviour than other well-known EAs [8], [9].

2 Differential Evolution

Differential Evolution (DE) is an effective and efficient method which has been proposed to solve optimization problems in continuous search spaces [8], [9]. The main advantage of DE is its simple structure, because it has few control variables, and it uses common concepts of Evolutionary Algorithms (EA). DE¹ is based on a population of individuals, generated through a similar operation to the classical *mutation*, where, for each individual x_i^t , a new individual y_i^{t+1} is generated as follows: $y_i^{t+1} = x_j^t + F(x_k^t - x_l^t)$, and F is called *scaling factor*. To have high diversity into the population, DE uses a *crossover* operator, between x_i^t and y_i^{t+1} , to generate the offspring x_i^{t+1} . Finally, the offspring is evaluated and it replaces its parent if its fitness is better than its parent (selection process).

There are several variants of DE, depending on the the selection for mutation operator, and the crossover scheme used. To distinguish its several variants we use the notation $a/b/c$, where "a" denotes the way an individual is selected to be mutated (random or best individual); "b" is the number of pairs of solutions chosen and "c" represent the crossover scheme used (binomial or exponential). Therefore, using this notation we have the following variants [6]: *rand/1/bin*, *rand/1/exp*, *best/1/bin* and *best/1/exp*; *current-to-rand/1* and *current-to-best/1*, which use an arithmetic recombination; *current-to-rand/1/bin*, which presents a combined discrete-arithmetic recombination; and *rand/2/dir*, which includes fitness function information to the mutation and recombination operators.

3 Immune Algorithms

The immune algorithms are inspired by the human's clonal selection principle, which suggests that among all possible cells, B lymphocytes, with different receptors circulating in the host organism, only those who are actually able to recognize the antigen will start to proliferate by cloning.

The proposed algorithm is population based, like any typical evolutionary algorithm and it is based on two entity types: antigens (Ag) and B cells receptor. The Ag's represent the problem to tackle, i.e. the function to optimize, whereas

¹ for a deeply knowledge on DE see [8], [9].

the B cells are a population of points of the search space of the given function. At each time step t the algorithm presents a population $P^{(t)}$ of size d . The initial population of candidate solutions at time $t = 0$ is randomly generated using uniform distribution in the relative domains of each function. The proposed algorithm generates each gene $x_i = B_{l_i} + \beta \cdot (B_{u_i} - B_{l_i})$, where $\beta \in [0, 1]$ is a random value, B_{l_i} and B_{u_i} are the lower and upper bounds of the real coded variable x_i respectively.

The function $Evaluate(P^{(t)})$ (see Table 3) computes the fitness function value of each B cell $\mathbf{x} \in P^{(t)}$. The evolution cycle ends when the maximum number of fitness function evaluations, T_{max} , is reached.

We used the classical cloning operator, which clones each B cell dup times producing an intermediate population $P^{(clo)}$, assigning to each clone the same age of its parent. The age of B cells, determines their life span into the population: when a B cell reaches the maximum allowed age, it dies, i.e. it is eliminated from the population. Subsequently, if a cloned B cell undergoes a successfully mutation (called *constructive mutation*), i.e. its fitness value has increased, it will be considered to have age equal to 0. Such a scheme, as proposed in [2], intends to give an equal opportunity to each new B cell to effectively explore the search landscape.

The hypermutation operators act on the B cell receptor of $P^{(clo)}$. We used the inversely proportional hypermutation operator, which tries to mutate each B cell receptor M times without the explicit usage of a mutation probability. The feature of this operator is that the number of mutations is inversely proportional to the fitness value, i.e. as the fitness function value of the current B cell increases, the number of mutations performed decreases. The mutation potential used was $\alpha = e^{(-\rho \cdot f)}$, where α represents the mutation rate and f is the fitness function value normalized in $[0, 1]$. The perturbation operator choose randomly a variable x_i , with $i \in \{1, \dots, \ell\}$ (ℓ is the length of B cell) and replace it with $x_i^{new} = ((1 - \beta) \cdot x_i) + (\beta \cdot x_{random})$, where $x_{random} \neq x_i$ is a randomly chosen variable and $\beta \in [0, 1]$ is a random number obtained with uniform distribution. As proposed in [1], the proposed algorithm does not use any additional information concerning the problem. For example the global optima is not considered when normalizing the fitness function value, but the best current fitness value decreased of a threshold θ .

The aging operator, used by the algorithm, eliminates old B cells, in the populations $P^{(t)}$, and $P^{(hyp)}$, maintaining high diversity in the current population, in order to avoid premature convergence. The maximum number of generations the B cells are allowed to remain in the population is determined by the τ_B parameter: when a B cell is $\tau_B + 1$ old it is erased from the current population, independently from its fitness value. The algorithm makes only one exception: when generating a new population the selection mechanism does not allow the elimination of the B cell with the best fitness function value (*elitist aging*). After the application of the immune operators the best surviving B cells are selected from the populations $P_a^{(t)}$ and $P_a^{(hyp)}$. In this way, the new population $P^{(t+1)}$, of d B cells, for the next generation $t + 1$, is obtained. If only $d' < d$ B cells have

Table 1. Pseudo-code of the proposed Immune Algorithm

| |
|--|
| <pre> Immune Algorithm($d, dup, \rho, \tau_B, T_{max}$) $FFE \leftarrow 0$; $N_c \leftarrow d \cdot dup$; $t \leftarrow 0$; $P^{(t)} \leftarrow \text{Init_Population}(d)$; Evaluate($P^{(t)}$); $FFE \leftarrow FFE + d$; while ($FFE < T_{max}$)do $P^{(clo)} \leftarrow \text{Cloning}(P^{(t)}, dup)$; $P^{(hyp)} \leftarrow \text{Hypermutation}(P^{(clo)}, \rho)$; Evaluate($P^{(hyp)}$); $FFE \leftarrow FFE + N_c$; ($P_a^{(t)}, P_a^{(hyp)}$) = $\text{Aging}(P^{(t)}, P^{(hyp)}, \tau_B)$; $P^{(t+1)} \leftarrow (\mu + \lambda)\text{-Selection}(P_a^{(t)}, P_a^{(hyp)})$; $t \leftarrow t + 1$; end_while </pre> |
|--|

survived, the $(\mu + \lambda)$ -*Selection operator* randomly selects $d - d'$ “old” B cells from $P_a^{(t)} \sqcup P_a^{(hyp)}$.

In Table 3 is showed the pseudo-code of the proposed immune algorithm.

4 Experimental Results

To better understand the search ability of the proposed IA and its real performances, we used a large set of experiments on two different categories of functions with different features, using high different dimensional values. Moreover, we compared our algorithm to several DE variants, as proposed in [6], [5], [4], to evaluate the goodness of the proposed solutions by IA. We used the first 13 well-known benchmark functions from [3]: *unimodal functions*, that are relatively easy to optimize, but their difficulty increases as the dimensional space increase, and *multimodal functions*, with many local minima, that represent the most difficult class of problems for many optimization algorithms. Last category is the most important, because the quality of the final results reflects the ability of the given algorithm to *escape* from local optima.

For all experiments we used the following values for IA: $d = 100$, $dup = 2$, $\tau_B = 15$, and $\theta = 75\%$, where θ represents the threshold used to decrease the best fitness function value obtained in each generation to normalize the fitness in the range $[0, 1]$ [1]. Moreover, we used $\rho = 7$ for high dimension values, whilst lower ρ values for smaller dimensions.

Figure 1 shows the mutation number obtained at different fitness function values, from worst to best. These experiment were obtained using small and high dimensional values. In the inset plot we give a zoom reducing the normalized fitness function in the range $[0.4, 1]$.

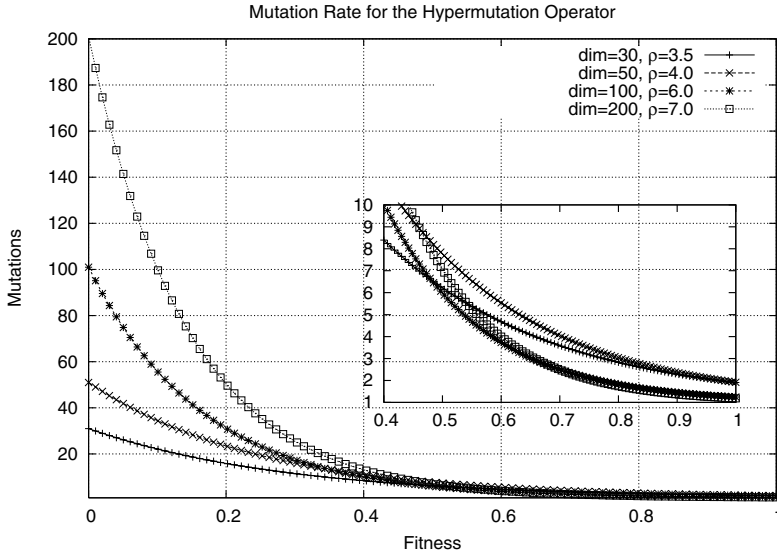


Fig. 1. Mutation Potential behavior using different dimension values

At each generation we computed the *mean value* of the best fit individuals for all independently runs and the *standard deviation*, to indicate the consistency of the algorithm.

In the first experiment, IA is compared with the 8 DE variants, proposed in [6], where T_{max} was fixed to 12×10^4 . For each function 100 independent runs were performed. The dimension of the functions was fixed to 30, and the results are shown in Table 2. Because in [6] the authors have modified the function f_8 to have its minimum at zero (rather than -12569.5), it is not included into the same table. The best results obtained by each algorithm are shown in boldface. From this table, one can see that IA outperforms the majority of the DE variants, either in unimodal class or multimodal, and it is comparable with the best DE algorithm.

In Table 3 IA is compared to *rand/1/bin* variant, using a different experimental protocol, proposed in [5]. For each experiment the maximum number of fitness function evaluations T_{max} was fixed to 5×10^5 , for function dimension 30 or less, whilst using 100 dimension, T_{max} was set to 5×10^6 . For each benchmark function, 30 independent runs were performed. For each function the mean best function values found in the last generation is shown in the first row, and standard deviation in the second. The results obtained for both 30 and 100 variables IA are comparable to the results obtained by *rand/1/bin*. In this comparison all results below 10^{-20} were reported as 0.0.

Recent developments in EAs field, have shown that to tackle complex search spaces, pure genetic algorithms (GA) need to use local search operators and specialized crossover [7]. In [4] two memetic versions of DE, which used *crossover based local search* (XLS), were proposed. As a last experiment, IA was compared

Table 2. Immune Algorithm versus DE variants on the unimodal and multimodal functions, using 30 dimension

| <i>Unimodal Functions</i> | | | | | | |
|------------------------------|--------------|------------|------------|------------|------------|------------|
| | f_1 | f_2 | f_3 | f_4 | f_6 | f_7 |
| IA | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0000489 |
| <i>rand/1/bin</i> | 0.0 | 0.0 | 0.02 | 1.9521 | 0.0 | 0.0 |
| <i>rand/1/exp</i> | 0.0 | 0.0 | 0.0 | 3.7584 | 0.84 | 0.0 |
| <i>best/1/bin</i> | 0.0 | 0.0 | 0.0 | 0.0017 | 0.0 | 0.0 |
| <i>best/1/exp</i> | 407.972 | 3.291 | 10.6078 | 1.701872 | 2737.8458 | 0.070545 |
| <i>current-to-best/1</i> | 0.54148 | 4.842 | 0.471730 | 4.2337 | 1.394 | 0.0 |
| <i>current-to-rand/1</i> | 0.69966 | 3.503 | 0.903563 | 3.298563 | 1.767 | 0.0 |
| <i>current-to-rand/1/bin</i> | 0.0 | 0.0 | 0.000232 | 0.149514 | 0.0 | 0.0 |
| <i>rand/2/dir</i> | 0.0 | 0.0 | 30.112881 | 0.044199 | 0.0 | 0.0 |
| <i>Multimodal Functions</i> | | | | | | |
| | f_5 | f_9 | f_{10} | f_{11} | f_{12} | f_{13} |
| IA | 11.69 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| <i>rand/1/bin</i> | 19.578 | 0.0 | 0.0 | 0.001117 | 0.0 | 0.0 |
| <i>rand/1/exp</i> | 6.696 | 97.753938 | 0.080037 | 0.000075 | 0.0 | 0.0 |
| <i>best/1/bin</i> | 30.39087 | 0.0 | 0.0 | 0.000722 | 0.0 | 0.000226 |
| <i>best/1/exp</i> | 132621.5 | 40.003971 | 9.3961 | 5.9278 | 1293.0262 | 2584.85 |
| <i>current-to-best/1</i> | 30.984666 | 98.205432 | 0.270788 | 0.219391 | 0.891301 | 0.038622 |
| <i>current-to-rand/1</i> | 31.702063 | 92.263070 | 0.164786 | 0.184920 | 0.464829 | 5.169196 |
| <i>current-to-rand/1/bin</i> | 24.260535 | 0.0 | 0.0 | 0.0 | 0.001007 | 0.000114 |
| <i>rand/2/dir</i> | 30.654916 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |

Table 3. Performance Comparison among IA and "rand/1/bin" on 13 functions benchmarks, using 30 and 100 dimensions

| | <i>30 dimension</i> | | <i>100 dimension</i> | |
|----------|---|--|--|---|
| | IA | <i>rand/1/bin</i> | IA | <i>rand/1/bin</i> |
| f_1 | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 |
| f_2 | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 |
| f_3 | 0.0 0.0 | 2.02×10^{-9} 8.26×10^{-10} | 0.0 0.0 | 5.87×10^{-10} 1.83×10^{-10} |
| f_4 | 0.0 0.0 | 3.85×10^{-8} 9.17×10^{-9} | 6.447×10^{-7} 3.338×10^{-6} | 1.128×10^{-9} 1.42×10^{-10} |
| f_5 | 12 13.22 | 0.0 0.0 | 74.99 38.99 | 0.0 0.0 |
| f_6 | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 |
| f_7 | 1.521×10^{-5} 2.05×10^{-5} | 4.939×10^{-3} 1.13×10^{-3} | 1.59×10^{-5} 3.61×10^{-5} | 7.664×10^{-3} 6.58×10^{-4} |
| f_8 | $-1.256041 \times 10^{+4}$ 25.912 | $-1.256948 \times 10^{+4}$ 2.3×10^{-4} | $-4.16 \times 10^{+4}$ $2.06 \times 10^{+2}$ | $-4.1898 \times 10^{+4}$ 1.06×10^{-3} |
| f_9 | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 |
| f_{10} | 0.0 0.0 | -1.19×10^{-15} 7.03×10^{-16} | 0.0 0.0 | 8.023×10^{-15} 1.74×10^{-15} |
| f_{11} | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 | 5.42×10^{-20} 5.42×10^{-20} |
| f_{12} | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 | 0.0 0.0 |
| f_{13} | 0.0 0.0 | -1.142824 4.45×10^{-8} | 0.0 0.0 | -1.142824 2.74×10^{-8} |

Table 4. IA versus *rand/1/exp*, *best/1/exp* and their memetic versions by [4], with $n = 50$ dimensional search space

| | IA | <i>rand/1/exp</i> | <i>best/1/exp</i> | DEfrDE | DEfrSPX |
|----------|---------------|---|---|---|---|
| f_1 | 0 ± 0 | 0 ± 0 0 ± 0 0.0535 ± 0.0520 | 309.74 ± 481.05 0 ± 0 0.0027 ± 0.0013 | 0 ± 0 0 ± 0 0.0026 ± 0.0023 | 0 ± 0 0 ± 0 $1 \cdot 10^{-4} \pm 4.75 \cdot 10^{-5}$ |
| f_5 | 30 ± 21.7 | 79.8921 ± 102.611 52.4066 ± 19.9109 90.0213 ± 33.8734 | $3.69 \cdot 10^{+5} \pm 5.011 \cdot 10^{+5}$ 54.5985 ± 25.6652 58.1931 ± 9.4289 | 72.0242 ± 47.1958 53.1894 ± 26.1913 66.9674 ± 23.7196 | 65.8951 ± 37.8933 45.8367 ± 10.2518 52.0033 ± 13.6881 |
| f_9 | 0 ± 0 | 0 ± 0 0 ± 0 0 ± 0 | 0.61256 ± 1.1988 0 ± 0 0 ± 0 | 0 ± 0 0 ± 0 0 ± 0 | 0 ± 0 0 ± 0 0 ± 0 |
| f_{10} | 0 ± 0 | $9.36 \cdot 10^{-6} \pm 3.67 \cdot 10^{-6}$ 0.0104 ± 0.0015 | $6.85 \cdot 10^{-6} \pm 6.06 \cdot 10^{-6}$ 0.0067 ± 0.0015 | $2.28 \cdot 10^{-5} \pm 1.45 \cdot 10^{-5}$ 0.0060 ± 0.0015 | $3.0 \cdot 10^{-6} \pm 1.07 \cdot 10^{-6}$ $0.0019 \pm 4.32 \cdot 10^{-4}$ |
| f_{11} | 0 ± 0 | 0 ± 0 $9.95 \cdot 10^{-7} \pm 4.3 \cdot 10^{-7}$ 0.0053 ± 0.010 | 0.1651 ± 0.2133 0 ± 0 0.0012 ± 0.0028 | 0 ± 0 0 ± 0 $4.96 \cdot 10^{-4} \pm 6.68 \cdot 10^{-4}$ | 0 ± 0 0 ± 0 $5.27 \cdot 10^{-4} \pm 0.0013$ |

Table 5. IA versus *rand/1/exp*, *best/1/exp* and their memetic versions by [4], with $n = 100$ dimensional search space

| | IA | <i>rand/1/exp</i> | <i>best/1/exp</i> | DEfrDE | DEfrSPX |
|----------|-------------------|--|---|--|--|
| f_1 | 0 ± 0 | $1.58 \cdot 10^{-6} \pm 3.75 \cdot 10^{-6}$ 59.926 ± 16.574 2496.82 ± 246.55 | 0.0046 ± 0.0247 30.242 ± 5.93 1729.40 ± 172.28 | 0 ± 0 11.734 ± 5.0574 358.57 ± 108.12 | 0 ± 0 1.2614 ± 0.4581 104.986 ± 22.549 |
| f_5 | 85.6 ± 31.758 | 120.917 ± 41.8753 12312.16 ± 3981.44 $3.165 \cdot 10^{+6} \pm 6.052 \cdot 10^{+5}$ | 178.465 ± 60.938 7463.633 ± 2631.92 $1.798 \cdot 10^{+6} \pm 3.304 \cdot 10^{+5}$ | 107.5604 ± 28.2529 2923.108 ± 1521.085 $2.822 \cdot 10^{+5} \pm 3.012 \cdot 10^{+5}$ | 99.1086 ± 18.5735 732.85 ± 142.22 16621.32 ± 6400.43 |
| f_9 | 0 ± 0 | 2.6384 ± 0.7977 234.588 ± 13.662 | 0.7585 ± 0.2524 198.079 ± 18.947 | 0.1534 ± 0.1240 17.133 ± 7.958 | 0.0094 ± 0.0068 27.0537 ± 20.889 |
| f_{10} | 0 ± 0 | $1.02 \cdot 10^{-6} \pm 1.6 \cdot 10^{-7}$ 1.6761 ± 0.0819 7.7335 ± 0.1584 | $9.5 \cdot 10^{-7} \pm 1.1 \cdot 10^{-7}$ 1.2202 ± 0.0965 6.7251 ± 0.1373 | $1.2 \cdot 10^{-6} \pm 6.07 \cdot 10^{-7}$ 0.5340 ± 0.1101 3.7515 ± 0.2773 | 0 ± 0 0.3695 ± 0.0734 3.4528 ± 0.1797 |
| f_{11} | 0 ± 0 | 0 ± 0 1.1316 ± 0.0124 20.037 ± 0.9614 | 0 ± 0 1.0530 ± 0.0100 13.068 ± 0.8876 | 0 ± 0 0.7725 ± 0.1008 3.7439 ± 0.7651 | 0 ± 0 0.5433 ± 0.1331 2.2186 ± 0.3010 |

Table 6. IA versus *rand/1/exp*, *best/1/exp* and their memetic versions by [4], with $n = 200$ dimensional search space

| | IA | <i>rand/1/exp</i> | <i>best/1/exp</i> | DEfrDE | DEfrSPX |
|----------|------------------|---|---|--|--|
| f_1 | 0 ± 0 | 50.005 ± 16.376 $5.45 \cdot 10^{+4} \pm 2605.73$ $1.82 \cdot 10^{+5} \pm 6785.18$ | 26.581 ± 7.4714 $4.84 \cdot 10^{+4} \pm 1891.24$ $1.74 \cdot 10^{+5} \pm 6119.01$ | 17.678 ± 9.483 9056.0 ± 1840.45 44090.5 ± 6122.35 | 0.8568 ± 0.2563 2782.32 ± 335.69 9850.45 ± 1729.9 |
| f_5 | 165.1 ± 71.2 | 9370.17 ± 3671.11 $4.22 \cdot 10^{+8} \pm 3.04 \cdot 10^{+7}$ $3.29 \cdot 10^{+9} \pm 2.12 \cdot 10^{+8}$ | 6725.48 ± 1915.38 $3.54 \cdot 10^{+8} \pm 3.54 \cdot 10^{+7}$ $3.12 \cdot 10^{+9} \pm 1.65 \cdot 10^{+8}$ | 5302.79 ± 2363.74 $2.39 \cdot 10^{+7} \pm 6.379 \cdot 10^{+6}$ $3.48 \cdot 10^{+8} \pm 1.75 \cdot 10^{+8}$ | 996.69 ± 128.483 $1.19 \cdot 10^{+6} \pm 4.10 \cdot 10^{+5}$ $1.21 \cdot 10^{+7} \pm 4.73 \cdot 10^{+6}$ |
| f_9 | 0 ± 0 | 0.4245 ± 0.2905 1878.61 ± 60.298 5471.35 ± 239.67 | 0.2255 ± 0.1051 1761.55 ± 43.3824 5094.97 ± 182.77 | 0.1453 ± 0.2771 352.93 ± 46.11 1193.83 ± 145.477 | 0.0024 ± 0.0011 369.88 ± 136.87 859.03 ± 99.76 |
| f_{10} | 0 ± 0 | 0.5208 ± 0.0870 15.917 ± 0.1209 19.253 ± 0.0698 | 0.4322 ± 0.0427 15.46 ± 0.1205 19.138 ± 0.0772 | 0.3123 ± 0.0426 9.2373 ± 0.4785 14.309 ± 0.3706 | 0.1589 ± 0.0207 6.6861 ± 0.3286 9.4114 ± 0.4581 |
| f_{11} | 0 ± 0 | 0.7687 ± 0.0768 490.29 ± 21.225 1657.93 ± 47.142 | 0.5707 ± 0.0651 441.97 ± 15.877 1572.51 ± 53.611 | 0.5984 ± 0.1419 78.692 ± 11.766 368.90 ± 41.116 | 0.1631 ± 0.0314 28.245 ± 4.605 85.176 ± 12.824 |

with *rand/1/exp*, *best/1/exp* and their memetic versions, called DEfrDE and DEfrSPX [4], respectively. For each experiment, the maximum number of fitness function evaluations T_{max} was fixed to 5×10^5 , and 50, 100 and 200 dimension variables were used. For each instance 30 independent runs were performed. The same functions proposed in [4]: f_1, f_5, f_9, f_{10} and f_{11} , were used. In Tables [4, 5] and [6], for the two DE variants and their memetic versions, are reported the results obtained varying the population size, with $n, 5n$ and $10n$, where n is the dimensional search space, as showed in [4]. Moreover, for each algorithm, is showed the *mean value* of the best fit individuals and *standard deviation*

($mean \pm sd$), used to indicate the consistency of the algorithm. In Table 4 we report the results obtained using $n = 50$ dimensional search space. From this table one can see that IA is comparable and in many cases outperforms DE and its memetic variants, especially for f_5 .

In Tables 5 and 6 one can see the results obtained using 100 and 200 dimensional search space. From these two tables it is clear that IA outperforms the compared algorithms when increasing the function dimension. It is important to highlight that our proposed algorithm outperforms DE variants and their memetic versions, in each function using smaller population size for $n = 100$ and $n = 200$.

5 Conclusion

The main features of the IA can be summarized as: (1) the *cloning* operator, which explores the neighbourhood of a given solution, (2) the *inversely proportional hypermutation* operator, that perturbs each candidate solution as a function of its fitness function (inversely proportional), and (3) the *aging operator*, that eliminates the oldest candidate solutions from the current population in order to introduce diversity and thus avoiding local minima during the search process.

In this research paper we presented an extensive comparative study illustrating the performance of a well-known immune algorithm [1], with the features mentioned above, and that of several differential evolution variants [6, 5] and their memetic versions [4]. We used the 13 classical benchmark functions from [3] (unimodal and multimodal functions) for our experiments. Furthermore the dimensionality of the problems was varied from $n = 30$ to $n = 200$ dimensions.

Our results suggest that the proposed immune algorithm is an effective numerical optimization algorithm (in terms of solution quality) particularly for the most challenging highly dimensional search spaces. In particular, increasing the dimension of the solutions space improves the performances of IA. Finally, the experimental results also show that the IA is comparable, and it often outperforms, all 8 DE variants as well as their memetic counterparts.

References

1. Cutello V., Nicosia G., Pavone M., Narzisi G.; "Real Coded Clonal Selection Algorithm for Unconstrained Global Numerical Optimization using a Hybrid Inversely Proportional Hypermutation Operator," in 21st Annual ACM Symposium on Applied Computing (SAC), vol. 2, pp. 950–954 (2006).
2. Cutello V., Morelli G., Nicosia G., Pavone M.; "Immune Algorithms with Aging Operators for the String Folding Problem and the Protein Folding Problem," in 5th European Conference on Computation in Combinatorial Optimization (EvoCOP), pp. 80–90 (2005).
3. Yao X., Liu Y., Lin G. M.; "Evolutionary programming made faster," IEEE Transaction on Evolutionary Computation, vol. 3, no. 2, pp. 82–102 (1999).
4. Noman N., Iba H.; "Enhancing Differential Evolution Performance with Local Search for High Dimensional Function Optimization," in Genetic and Evolutionary Computation Conference (GECCO), pp. 967–974 (2005).

5. Versterstrøm, J., Thomsen R.: " *A Comparative Study of Differential Evolution, Particle Swarm Optimization, and Evolutionary Algorithms on Numerical Benchmark Problems,*" in Congress on Evolutionary Computing (CEC), vol. 1, pp. 1980-1987, (2004).
6. Mezura-Montes E., Velázquez-Reyes J., Coello Coello C.: " *A Comparative Study of Differential Evolution Variants for Global Optimization,*" in Genetic and Evolutionary Computation Conference (GECCO), vol. 1, pp. 485-492 (2006).
7. Goldberg D. E., Voessner S.: " *Optimizing Global-Local Search Hybrids,*" in Genetic and Evolutionary Computation Conference (GECCO), pp. 220-228 (1999).
8. Storn R., Price K. V.: " *Differential Evolution a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces,*" in Journal of Global Optimization, vol. 11, no. 4, pp. 341-359 (1997).
9. Price K. V., Storn M., Lampien J. A.: " *Differential Evolution: A Practical Approach to Global Optimization,*" Springer-Verlag (2005).

Self-adaptive Evolutionary Methods in Designing Skeletal Structures

Adam Borkowski¹ and Piotr Nikodem²

¹ Institute of Fundamental Technological Research,
Warsaw, Poland

abork@ippt.gov.pl

² pnikodem@poczta.onet.pl

Abstract. This paper focuses on modified genetic algorithm based on the graph representation and specialized genetic operators. Advantages of changed representation, multi-level organization as well as self-adaptive aspects of the proposed method are described. The results of algorithm usage in optimising skeletal structures are also presented.

1 Introduction

The aim of our work is to develop a tool for supporting the design of skeletal structures. An important part of such design is the optimisation process, which has to reduce structural cost, most commonly in terms of the overall construction weight. The search for optimum solution can be performed at three levels: a topological level, a geometrical level and a component level.

At the last level only the parameters of individual structural components (i.e. the cross-sections of the bars) are optimized. The geometrical level is responsible for proper positioning of the components, which means finding optimal coordinates of the structural nodes. At the highest, topological level, the number of components and the connections between them are considered.

The optimum design of skeletal structures is a widely explored research area. The procedures for the geometric and component levels are well described, while the topological optimisation still remains an open question. Certain progress has been achieved by using the topological derivative [1], the theory of homogenisation [2], the bubble evolution method [3]. Usually the search is performed by means of the genetic algorithm, since classical optimisation methods are not applicable.

In our model a graph-based representation of skeletal structure is used. Graphs have much more complex internal structure than binary strings. This is obvious advantage in terms of the expression power. From the other hand, it requires the development of new operators for evolutionary computing.

The required operators are defined within a concept of hierarchical graph - a graph in which nodes can contain internal nodes. A crossover operator can be basically described as exchanging nodes on some level of hierarchy between two graphs. Additionally, to ensure that this operator produces meaningful graphs, a notion of context similarity (homology) is used. A mutation is defined as a change

of the graph structure (removing or adding nodes or edges) or a modification of labels and attributes assigned to the graph nodes and edges.

Based on fundamentals related to hierarchical graphs and operators defined on them, a customised evolutionary algorithm is built. This algorithm is organized as a sequence of the following steps: selection, crossover, mutation and evaluation. The optimised skeletal structure is evaluated with the assistance of commercial CAD tool (Robot Millennium) [4]. This tool provides internal forces, reactions and displacements induced by given load. It allows us also to find dimensions of structural components that ensure the fulfilment of the Code of Practice.

The procedure described above has been implemented in the Java programming language and a number of numerical experiments were performed. We took the well-known cantilever truss as the benchmark problem. Three groups of experiments were done: a decoupled optimisation on each level, a full-scale optimisation under user control and a full-scale optimisation in self-adaptive mode.

The full-scale optimisation begins with the topology optimisation. When the satisfying solution is found, it is used as a seed for generating an initial population in the geometrical optimisation. After promising result at the geometrical level is found, it is used similarly for starting the process of component optimisation. The full-scale optimisation process can be controlled either by the user or by the self-adaptive software module. In both cases the parameters of the evolutionary algorithm can be changed during the optimisation process in order to improve convergence.

2 Representation of Structures

One of the most important things in whole design/optimization task is representation of a considered artefact. The chosen model has to fulfil several requirements: it has to have adequate expression power, considering computer support it has to be simple enough to enable automatic processing and it should provide reusability if possible. These requirements are partially contradicting, causing sometimes serious problems at early stages of analysis.

In the classical genetic algorithm binary strings are used for coding the design. This coding showed many advantages and disadvantages as well. To the most important advantages should be qualified: simplicity, well defined and reusable operators and theoretical fundamentals. From the other side in complicated tasks binary strings exceeded reasonable lengths and caused problems with the convergence of algorithm. To avoid those problems real-number encoding in form of one- or two-dimensional table was proposed. It proved its usability for a group of tasks, but also copied disadvantages from binary encoding. Long-term solution is the usage of more complicated structures: trees or graphs. In our work hierarchical graphs are used, since they provide very good expression power and good theoretical foundations as well.

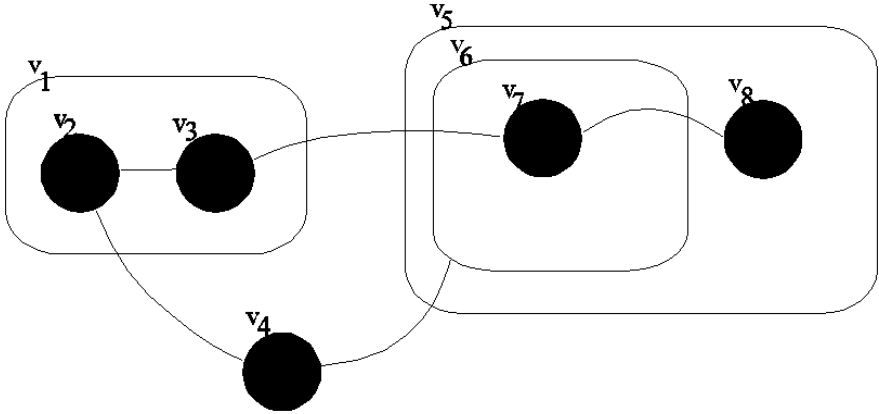


Fig. 1. Example of hierarchical CP-graph. Bonds are not presented for clarity.

Hierarchical CP-graph is an extension to CP-graph. While CP-graph is able to represent relations between components, hCP-graph provides additional mechanism for hierarchical dependencies. It allows the designer to look at the object being designed at different levels of abstraction and comfortably separate different stages of design process. Depending on a current design stage a more detailed view down to the level of subcomponents with relation between them or only the highest level of the structure without too many details may be chosen.

Basically speaking hierarchical CP-graphs are CP-graphs in which nodes can contain internal nodes. These nodes, called children also can contain other internal nodes. Each node has at most one direct ancestor, node which has no direct ancestor is said to be at the highest level of hierarchy. To each node a set of bonds is assigned. An edge is defined as a pair of bonds, thus two nodes are connected not directly, but through its bonds. At most one edge can be connected to the bond. Nodes at the different levels can be connected by the edge, only connection between node and its ancestor is forbidden. Nodes and edges are labeled and attributed (by means of node and edge labeling and attributing functions respectively). Attributes may represent geometrical properties (size, coordinates), material characteristic or relation parameters.

Graphs represent structure of designed artifacts, which is sufficient at early stages of design process. To provide mechanism for object description in final stages of design process concepts of graph implementation and interpretation were proposed. Implemented graph is a graph in which value of all attributes is determined. Implemented graph represent exactly one real design and can be in deterministic way interpreted. Interpretation is defined as a pair of functions, which first assign geometrical objects to graph nodes and second establishes proper relations between those objects according to knowledge encoded in graph edges. More information about CP-graphs and h-CP graphs is provided in [5], [6], [7]. An example of the hierarchical CP-graph is presented in Fig. 1.

3 Evolutionary Methods

Intelligent methods are very closely coupled with the representation of data. As hierarchical CP-graph was chosen as a data model a new set of 'genetic' operators have to be designed for them, as standard genetic operators originally designed for binary strings are not applicable anymore. Graphs have much more complex internal structure than binary string, which require completely different operators to enable construction of proper evolutionary process.

In binary strings meaning of a bit was defined by its position. Thus during crossover operation exchanging appropriate parts of strings gives as a result correct binary string, which represent valid design. To achieve same within graph structures, kind of similarity relation has to be defined. This relation, called homology is responsible for establishing homologous (having similar or identical function) subgraphs of selected graphs and thus selects appropriate parts of graph exchanged in the crossover process. This relation is defined, based on the indentment of a graph part, not on the actual internal structure or on values of attributes. Technically only labels of nodes will be taken into consideration.

The homology relation is defined on three levels, which differ in terms of requirements. Context free homology is the weakest of them and requires two subgraphs have the same number of nodes with identical labels at highest level of hierarchy. The second one, called strongly context dependent homology has same requirements as previous one extended with additional rule. Top-level nodes of subgraphs have to have identically labeled ancestors up to the highest level of hierarchy in parent graphs. Last one, weakly context dependent homology is placed between both previously described relations. It takes into consideration only direct ancestors of the subgraph top-level nodes. Extended description of homology relation can be found in [7]. Based on homology relation, crossover operator is defined.

In the mutation operator several subtypes can be distinguished. All operations in which graph structure is changed (adding and removing edges, adding and removing subgraphs) are coupled into the group called structural mutation. Second type, called value mutation is responsible for changing values of nodes and edges attributes. Simultaneous change of attributes values for a group of nodes is called macro-mutation.

Well-known selection algorithms can be used without any modifications. Fitness functions are strongly coupled with actual design problem. Examples of the crossover and the mutation operators as well as the selection algorithm and the fitness function used in our work will be provided in the next paragraph.

4 Practical Development

Based on fundamentals presented in previous paragraphs, a customised evolutionary algorithm for solving optimum skeletal structure problem was built. This algorithm uses the canonical sequence of genetic algorithm: selection, crossover, mutation, evaluation. This sequence is repeated until acceptable solution is found.

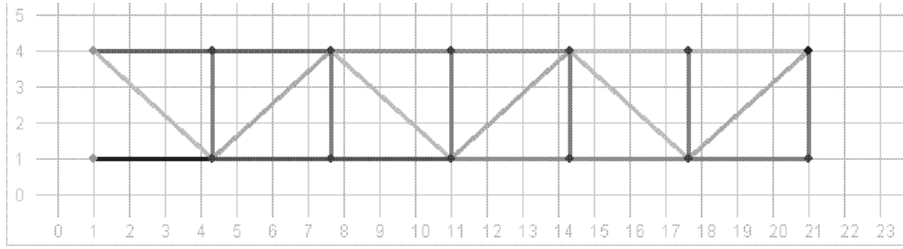


Fig. 2. Optimal result obtained in one of topology optimization experiments

For the selection process two well-known algorithms were tested: the roulette and the random choice without repeat. Several fitness functions were developed: most of them are based on summarizing forces in particular bars. To evaluate those forces external, professional commercial CAD tool called Robot Millennium was used. Besides internal forces this tool provides reactions, displacements induced by given load and performs and norm calculations as well. Best solution found in particular generation is copied to next generation without modification, which assures not losing the best solution over whole evolution process.

To enable practical experiments all described concepts and algorithms were implemented in the Java programming language. As a benchmark problem well-known 2D cantilever truss was chosen.

In the first experiment, the optimum topology search was performed under given nodal coordinates and given structural components. The cantilever is assembled from a number of panels (minimal and maximal number of them was configurable) of a certain type. We are looking for the best solution in the sense of the number and the type of panels. On the data structure level panels are encoded as nodes at highest level of hierarchy. The crossover is implemented as exchanging groups of panels between graphs, the mutation - as removing, adding or changing type of the panel (all operations belong to the type of structural mutation). For the fitness evaluation variety of functions of bars lengths and internal forces were used. The obtained optimum result is presented in Fig. 2.

During the experiment of optimum geometry search experiment, number of elements, connections between them and properties of bars were kept constant. Only positions of nodes were modified (minimum and maximum values for geometric coordinates were configurable). The crossover was defined as in the previous experiment, while the mutation changed values of nodal coordinates (nodes attributes - value mutation). The same fitness functions as in the previous experiment were used. The best result obtained during one of experiments is presented in Fig. 3.

In the next experiment - optimization of components - both the topology and the geometry were fixed and properties of the components were variable. The set of possible cross-sections was given a priori. We were looking for the optimal solution in the sense of cross-sections defined for particular bars. The crossover remains as in previous experiments, while the mutation changes values of material attributes assigned to nodes. As this change is performed simultaneous for

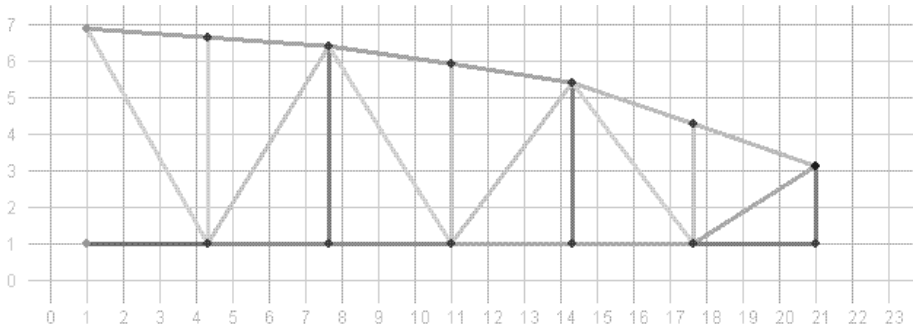


Fig. 3. Best result obtained in one of geometry optimization experiments

| Length (m) | Force Fx(N) | Section | Weight per unit | Weight (N) | Correct |
|------------|-------------|----------|-----------------|------------|---------|
| panel 1 | type 1 | | | | |
| 3.33 | 33,829.26 | RO 70x8 | 120.13 | 400.44 | true |
| 3.34 | -29,579.83 | RO 70x8 | 120.13 | 401.66 | true |
| 5.65 | -308.17 | RO 70x8 | 120.13 | 678.94 | true |
| 6.79 | -8,835.19 | RO 70x8 | 120.13 | 815.33 | true |
| panel 2 | type 2 | | | | |
| 6.37 | 9,394.09 | RO 108x8 | 193.19 | 1,230.27 | true |
| 3.33 | 24,572.75 | RO 108x8 | 193.19 | 643.97 | true |
| 5.43 | 0.00 | RO 108x8 | 193.19 | 1,048.27 | true |
| 3.34 | -29,557.43 | RO 108x8 | 193.19 | 645.44 | true |
| panel 3 | type 1 | | | | |
| 6.37 | -8,333.80 | RO 70x8 | 120.13 | 765.01 | true |
| 3.33 | 24,572.75 | RO 70x8 | 120.13 | 400.44 | true |
| 3.37 | -20,417.39 | RO 70x8 | 120.13 | 404.54 | true |
| 4.95 | 284.69 | RO 70x8 | 120.13 | 594.40 | true |
| panel 4 | type 2 | | | | |
| 3.33 | 15,073.35 | RO 108x8 | 193.19 | 643.97 | true |
| 3.37 | -20,459.76 | RO 108x8 | 193.19 | 651.91 | true |
| 4.42 | 0.00 | RO 108x8 | 193.19 | 854.45 | true |
| 5.54 | 8,535.31 | RO 108x8 | 193.19 | 1,069.94 | true |
| panel 5 | type 1 | | | | |
| 5.54 | -8,316.11 | RO 51x5 | 55.70 | 308.49 | true |
| 3.33 | 15,073.35 | RO 51x5 | 55.70 | 185.67 | true |
| 3.51 | -10,613.60 | RO 51x5 | 55.70 | 195.73 | true |
| 3.31 | 161.43 | RO 51x5 | 55.70 | 184.42 | true |
| panel 6 | type 2 | | | | |
| 3.96 | 11,973.05 | RO 51x5 | 55.70 | 220.80 | true |
| 3.53 | -10,665.79 | RO 51x5 | 55.70 | 196.69 | true |
| 3.33 | 0.00 | RO 51x5 | 55.70 | 185.67 | true |
| 2.15 | 0.00 | RO 51x5 | 55.70 | 119.50 | true |

Fig. 4. Optimal result obtained in one of components optimization experiments

the whole panel the macro-mutation is used. The fitness was measured as the overall structural weight. The optimum solution is shown in Fig. 4.

All described experiments were performed in manual and automatic mode. In the manual mode user set initial parameters for the genetic algorithm and begin optimization for specified number of generations. After that he can take decision if change some values of parameters and continue evolution for some additional number of generations. In automatic mode user also begins optimization process

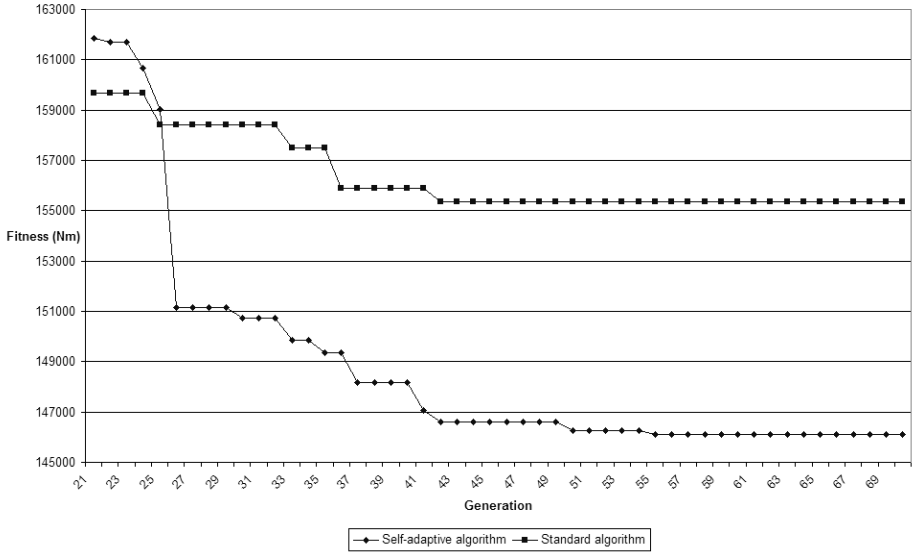


Fig. 5. Geometry optimization. Best individual fitness in dependence of generation number. Self-adaptive algorithm shows better convergence in late stages of process.

with setting initial parameters for the genetic algorithm. After that evolution is controlled by the independent software module. Changes of the genetic algorithm parameters and evolution stop are performed without any user interaction.

Changes of the genetic algorithm parameters during evolution process can significantly improve performance, especially in late stages. In the manual mode that changes requires constant user attention during statistics analysis and consumes big amount of user’s time. The automatic mode shows its advantage in this area: evolutions took less time (both user and general) as decisions are made faster and without user interaction.

The full-scale optimisation process groups all three stages of optimization. It begins with the topology optimization. Satisfying solution found in this process is used as a basis for generating an initial population in the geometrical optimisation. After promising result at the geometrical level is found, it is used similarly for starting the process of component optimisation. Process can be controlled by the user or by the software module. Despite some limitations of current implementation, experiments showed superiority of self-adaptive full-scale optimization to decoupled series of optimizations in sense of number of evaluations required to achieve similar results.

As a benchmark problem for the presentation, geometrical part of optimization was chosen. Series of experiments showed that self-adaptive algorithm gave about 4% better results than classical algorithm within same number of generations. Best individual fitness chart, based on data from sample experiment is presented in Fig. 5.

5 Conclusions and Future Work

We presented modified evolutionary algorithm defined on graph structures. It was used to support skeletal structure design/optimization. Several evolution strategies were tested. Strategy with dynamic changes of algorithm parameters during evolution showed its superiority over strategy without the changes. Independent software module was created to control changes during evolution and showed its usefulness. The full-scale, software-controlled optimisation confirmed its superiority over the decoupled manual-controlled mode.

Self-adaptive software module seems to be promising tool for evolution process control. In the future we intend to extend the software module to an independent agent that will control the whole optimisation process. This agent will not only tune parameters of the evolution process at the subsequent stages, but will be able to control several simultaneous populations on different levels and exchange individuals between them.

References

1. Bojczuk, D. and Mroz, Z.: Optimal topology and configuration design of trusses with stress and buckling constraints. *Journal of Structural Optimization* 17 (1999) 25-35
2. Bendsoe, M.: *Methods for the Optimization of Structural Topology, Shape and Material*. Berlin: Springer-Verlag (1995)
3. Eschenauer, H. and Schumacher, A.: Bubble method for topology and shape optimization of structures. *Journal of Structural Optimization* 8: (1994) 42-51
4. Robot Millennium version 18.0. User's Manual (2005)
5. Grabska, E.: Theoretical Concepts of Graphical Modelling. Part one: Realization of CP-graphs. *MG&V* 2(1) (1993) 3-38
6. Grabska, E.: Theoretical Concepts of Graphical Modelling. Part two: CP-graph Grammars and Languages. *MG&V* 2(2) (1993) 149-178
7. Strug, B.: Zastosowanie algorytmow ewolucyjnych i transformacji grafowych w projektowaniu graficznym wspomagany komputerowo. Doctoral dissertation (polish only). IPPT PAN (2000)

An Evolutionary Approach to Task Graph Scheduling

Saeed Parsa, Shahriar Lotfi, and Naser Lotfi

Faculty of Computer Engineering, Iran University of Science and Technology, Tehran, Iran
{parsa,shahriar_lotfi}@iust.ac.ir, naser1767@yahoo.com

Abstract. Effective scheduling is of great importance to parallel programming environments. The aim is to minimize the completion time of task graphs. The completion time of a task graph is directly affected by the length of its critical path. Hence, the trend of an evolutionary approach for task graph scheduling can be biased towards reduction of the critical path. In this paper, a new genetic scheduling algorithm is presented. The algorithm, in the first priority, minimizes the critical path length of the parallel program task graph and in the second priority minimizes the inter-processor communication time. Thereby, it achieves a better scheduling in comparison with the existing approaches.

1 Introduction

Efficient scheduling is of great importance to achieve higher performance in parallel programming environments. Scheduling must be performed in such a way that it could minimize the completion time of parallel program code considering the time required to perform the program tasks and inter-processors communications. There are six known deterministic approaches to solve the task graph scheduling problem: ETF [11], DLS [18], LAST [3], HLFET [5], ISH [13], MCP [20]. Amongst these approaches MCP is known as the most efficient scheduling algorithm [14] and it has been suggested as a basis to evaluate scheduling algorithms [14]. However, optimal scheduling of task graphs is a NP-hard problem [4]. Therefore, deterministic approaches are not recommended for scheduling of task graphs with a relatively large number of tasks [4]. Instead, an evolutionary approach such as genetic algorithms may be applied to solve such problems, effectively [6], [8], [10], [16], [19].

The representation of a problem solution, encoding scheme, highly affects the speed of genetic algorithms. In HAR [10] encoding scheme a chromosome consists of an ordered list of numbers each representing the priority or height of the task as a function of its longest distance from starting nodes of the task graph. In this scheme precedence relation between the tasks depends on the height of the tasks. Therefore, even if there is no relation between two tasks with different heights those two tasks can not be executed in parallel. Hence, the search space in HAR may not contain, in general, any optimal solution for the scheduling problem under consideration.

Solutions in CGL [6] approaches are represented as list of strings, each string corresponding to one processor of the target system. The strings maintain both the assignment and execution order of the tasks on each processor. However, this string representation which is similar to a sparse matrix entails a high amount of the main

storage and results in a crossing over operator which is relatively time consuming. Furthermore as described in [16], solutions generated by genetic operators do not generally resemble the solutions from which they are derived, a direct consequence of the rigidity of the string representation.

To reduce the complexity of genetic operators a two structure representation for the solutions which is suitable for task partitioning is proposed in an algorithm called BCGA [16]. A major difficulty with this approach is that the number of processors used for the scheduling is limited to the number of clusters. Therefore, if the number of clusters, n_{cl} , is less than the number of processors, n_{pr} , then $n_{pr} - n_{cl}$ processors will not be used for the scheduling. The solution representation in BCGA decomposes a schedule into two independent structures: a task-to-processor assignment matrix that stores the assignment of tasks to processors, and a topological-sort vector representing the execution order of the tasks in the schedule. This representation also disaffects the speed of the genetic algorithm. We have developed a simple chromosomal representation which, as shown in Sect. 2.1, results in a faster evolution compared with the above mentioned approaches, CGA and BCGA. In this representation, the order of allocation and execution of each task on parallel processors can be specified. Also the global order of the task execution in whole task graph is specified.

The performance of a genetic algorithm is often sensitive to the quality of its initial population. The better the fitness of the initial population, the lower the number of generations required to get the final individuals. As described in Sect. 2.2, by creating a three quarter of initial population as graph scheduling with near minimal critical path length the convergence rate of our proposed algorithm is highly accelerated. In PSGA [8], FGL [6] and CGL [6] approaches inter-tasks communication costs are not considered whilst we have considered the cost of communication in our approach.

Encoding schemes highly affect the performance of crossover operators. The encoding scheme presented in [19], has made it possible to have scheduling, resulted from applying crossover operator, with a same task assigned to several processors. We have applied a two-point crossover operator, which does not generate such a problem for scheduling of task graphs coded with our encoding scheme. As described in Sect. 2.5, since a two-point crossover operator often provides the highest power of search [7], a two-point crossing over operator is applied in our proposed algorithm.

The remaining parts of this paper are organized as follows: In Sect. 2, a new scheduling algorithm is proposed. Section 2.4 presents an algorithm to compute the fitness of individual chromosomes. In Sect. 2.5, the crossover and mutation operators, applied within the proposed algorithm, are described. Practical evaluations of the proposed scheduling algorithm are presented in Sect. 3.

2 Task Graph Scheduling Algorithm

In this section a pseudo code description of our genetic scheduling algorithm is presented. The input to the algorithm is a parallel code partitioned and represented as a directed acyclic graph (DAG), $G = (V, E)$, where V is a set of vertices or tasks of the parallel code and E is a set of directed edges between the tasks. Each directed edge represents the order of execution and the time required to communicate the tasks at the two ends of the edge. Apparently, the cost can be assumed zero when the tasks

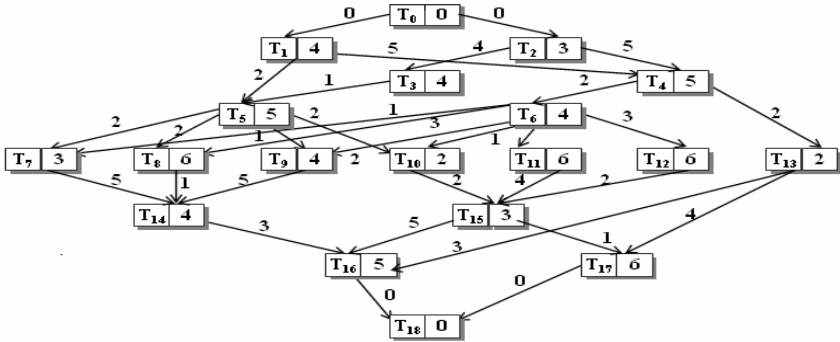


Fig. 1. A sample task graph

are executed on the same processor. Each vertex of the task graph is a pair representing the name and the execution time of the task. Below, in Fig. 1, a sample task graph [2] is presented.

The aim is to find an optimal scheduling for running task graph on multiprocessors such that the total completion time is minimized. We assume that processors are fully connected [19]. In a task graph, nodes that have no predecessors are nominated entry node, and nodes that have no successors, are nominated exit node. As described above in Sect. 1, task scheduling is an NP-hard optimization problem which can be best solved by applying a genetic optimization approach [4]

2.1 Chromosome Representation

It is recommended to represent a scheduling or chromosome as an ordered list of strings. Each string represents the execution order of the tasks assigned to a specific processor [16]. The problem is the global order of the tasks executions which is not shown in these representations of scheduling solutions [6]. To resolve the problem a new representation of schedules based on the string representation is given Fig. 2. In this new encoding scheme, a scheduling solution is represented as an array of pairs (T_i, P_j) where T_i indicates the task number assigned to the processor P_j . Using this representation both the execution order of the tasks on each processor and the global order of the tasks executions on the processors are defined in each chromosome.

As described above in the current string representations of scheduling solutions, the length of the strings representing the tasks assigned to each processor varies. This variation aggravates the complexity of the genetic scheduling algorithm. However, in the proposed representation, illustrated in Fig. 2, the length of the string representation of chromosomes are identical.

| | | | | | | | | | | | | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|----------|----------|----------|-------|----------|-------|-------|----------|----------|----------|----------|----------|
| T_0 | T_2 | T_3 | T_1 | T_4 | T_5 | T_6 | T_{11} | T_{13} | T_{12} | T_9 | T_{10} | T_8 | T_7 | T_{14} | T_{15} | T_{16} | T_{17} | T_{18} |
| P_0 | P_1 | P_2 | P_0 | P_1 | P_2 | P_1 | P_1 | P_1 | P_1 | P_0 | P_0 | P_0 | P_2 | P_0 | P_1 | P_0 | P_1 | P_1 |

Fig. 2. A chromosome created by scheduling the task graph in Fig. 1

2.2 Initial Population

The evolutionary process of genetic algorithms starts with an initial population. The quality of the initial population affects the performance of the genetic algorithm. Having an initial population with acceptable fitness values, the speed of reaching an optimal solution will be increased dramatically [9]. Further, high diversity in the population inhibits early convergence to a locally optimal solution. In order to maintain diversity, in our proposed scheduling algorithm, about a quarter of the individuals of the initial population are created randomly. To have an acceptable fitness, the rest of the initial population is created by applying a new approach described in this section.

Within a task graph the length of the critical path is equal to the earliest time to complete the scheduling. In our new approach to produce high quality chromosomes, the earliest start time for the tasks in a given task graph is calculated, firstly. The tasks are then sorted descending by earliest start time. A new algorithm for computing the earliest start time, $T_{tlevel}(n_i)$, for each task number, n_i , is described in Fig. 3, below. This algorithm also finds the nodes on the critical path. The idea is to minimize the length of critical path in order to generate scheduling with minimum completion time.

```

1: Initialize length, list of nodes and number of nodes on
   critical path to zero: LenghtOfCriticalPath :=
   NodesOnCriticalPath := NumberOfNodesOnCriticalPath := 0.
2: For each node  $n_i$  in  $V$  such that  $G = (V, E)$  Do
   Initialize  $T_{tlevel}(n_i)$ , NodesOnLongestPathTo( $n_i$ ) to  $\emptyset$ ;
   Initialize NumberOfNodesOnLongestPathTo( $n_i$ ) to 0.
3: For each node  $n_i$  in  $V$  such that  $G = (V, E)$  Do  $T_{tlevel}(n_i) := 0$ .
4: For each node  $n_i$  in  $V$  Do ReferenceCount( $n_i$ ) := # Parents( $n_i$ )
5: For each node with no parents Do Add the node to ready list
   ReadyList := {  $n_i \in V \mid$  ReferenceCount( $n_i$ ) = 0 }
6: While the ready list is not empty Do
   Select a node  $n_k$  from the ready list, randomly.
   For each child  $n_j$  of node  $n_k$  Do ReferenceCount( $n_j$ ) += -1.
   If ReferenceCount( $n_j$ ) = 0 Then Append  $n_j$  to the ready-list.
   Set the Earliest start time of  $n_j$ ,  $T_{tlevel}(n_j) = \text{Max}(T_{tlevel}(n_j,$ 
      $T_{tlevel}(n_k) + \text{ExecutionTime}(n_k) + \text{CommunicationCost}(n_k, n_j))$ 
   LenghtOfCriticalPath =
      $\text{Max}(\text{LenghtOfCriticalPath}, T_{tlevel}(n_j + \text{ExecutionTime}(n_j))$ 
   If  $T_{tlevel}(n_j) = T_{tlevel}(n_k) +$ 
      $\text{ExecutionTime}(n_k + \text{CommunicationCost}(n_k, n_j))$  then
     Copy NodesOnLongestPathTo( $n_k$ ) onto NodesOnLongestPathTo( $n_i$ ) .
     Append  $n_k$  to the end of the list NodesOnLongestPathTo( $n_i$ ) .
     Add 1 to NumberOfNodesOnLongestPathTo( $n_i$ ) .
   If  $T_{tlevel}(n_j) > \text{LenghtOfCriticalPath}$  Then
     LenghtOfCriticalPath :=  $T_{tlevel}(n_j)$ ;
     Copy NodesOnLongestPathTo( $n_i$ ) onto NodesOnCriticalPath.
     Put the Node  $n_i$  at the end of NodesOnCriticalPath.
     LenghtOfCriticalPath :=  $T_{tlevel}(n_j) + \text{ExecutionTime}(n_j)$  .
     Add 1 to NumberOfNodesOnCriticalPath.

```

Fig. 3. An algorithm to evaluate earliest start time and critical path

A reference count indicating the number of predecessors of a task which are not scheduled is assigned to each task within the task graph. The value of the reference count for the tasks with no immediate predecessors is zero. To assign tasks to processors, a task from within the tasks with reference count of zero is selected randomly. If the selected task is on the critical path then, in order to reduce the length of the critical path, the task will be assigned to the processor containing its parent on the critical path. Otherwise, to reduce the inter-task communication cost, the task will be assigned to the processor including tasks with proportionally higher number of communications with the task. After a task is assigned to a processor, the reference count of each task immediately after the task in the task graph is reduced by one and the task is removed from within the task list. The rest of the tasks are selected and processed as described above until there are no more tasks in the task list.

2.3 A New Scheduling Algorithm

Below in Fig. 4, a pseudo code description of our proposed algorithm is given.

```

1: Create initial population as described in Sect. 2.2:
   Produce a quarter of the population randomly.
   Produce the rest of the population as follows:
   Find the earliest start time (EST) for each task.
   Identify all the tasks on the critical path.
   Sort tasks according to EST in a linear list, ascending.
   Repeat Select randomly a task amongst the tasks ready to
       schedule at the beginning of the linear list.
       If the selected task is on critical path
       Then Assign the task to the processor including the
           previous task on critical path.
       Else Assign the task to the processor including a task
           with the highest interconnections with the task.
       Remove the selected task from the linear list.
   Until the linear list of tasks is empty.
2: While no. generations is less than a given number Do
   For each Chromosome in current population Do
       Calculate its fitness value as described in Sect. 2.4.
   Create intermediate generation as follows:
   Add the fittest chromosome to the intermediate population.
   Repeat Apply roulette wheel to select two chromosomes.
       Apply the crossover operator, described in Sect. 2.5.
       Apply the mutation operator, described in Sect. 2.5.
   Until the intermediate population size is completed.
   Copy the intermediate population over current population.

```

Fig. 4. A new genetic task graph scheduling algorithm

2.4 Computing Fitness of Chromosomes

The completion time of a scheduling represents its fitness. To estimate the fitness, a new algorithm is presented in Fig. 5. In this algorithm a local timer is assigned to each processor. The timer indicates the last time when a task assigned to a processor is expected to complete. Current time is kept in a global timer. If the value of the global

timer is more than the local timer of a processor P then the processor is idle and it may execute a task which is ready to execute and assigned to the processor. A task is ready to execute if its parent tasks in the task graph are all executed.

- 1: **For each** task, t , in the selected chromosome **Do**
 Set its reference count equal to the number of immediate parents of the task in the task graph.
- 2: **For each** processor P_i **Do** set its local timer, S_i , to zero.
- 3: Set the global timer S to zero.
- 4: Starting with the leftmost task T in the chromosome: **Repeat**
If referenceCount(T)#0 **Then** the chromosome is not acceptable
 Let P_i be the processor to which task T is assigned.
 Read the value of timer, S_i , of P_i from the chromosome.
If $S \geq S_i$ **Then** the processor is idle.
 Add sum of S and execution time, t , of the task, T , to S_i .
 Add one unit of time to the global timer, S .
Else If $S=S_i$ **Then** the processor has finished with the task.
 Reduce one from reference count of each child of the task.
 Select the next task T from the chromosome.
Until all the tasks are scheduled.
- 5: Set Fitness equal to the max(S_i) for all processors, P_i .
- 6: **For each** task T_j assigned to processor P_i **Do If** predecessors of T_j are not assigned to P_i **Then** add communication costs between T_j and each of its predecessors to the Fitness.
- 7: **Return** Fitness as the fitness of the chromosome.

Fig. 5. An algorithm to compute fitness value

2.5 Crossover and Mutation

To give chromosomes with a higher fitness a higher chance to parent new children during crossover, we select parents by applying a *Roulette Wheel* selection approach. The selected parents are combined by applying a two-point crossover operator. The cut points are selected randomly. Two new children are created by swapping the processors assigned to the tasks within the cut points in the parents. An example of the crossover operation is given in Fig. 6.

| | | | | | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|--------------|-------|-------|-------|-------|-------|-------|-------|
| Parent 1: | | | | | | | | Offspring 1: | | | | | | | |
| T_2 | T_3 | T_1 | T_4 | T_6 | T_8 | T_7 | T_5 | T_2 | T_3 | T_1 | T_4 | T_6 | T_8 | T_7 | T_5 |
| P_2 | P_1 | P_0 | P_1 | P_1 | P_0 | P_0 | P_2 | P_2 | P_1 | P_0 | P_1 | P_2 | P_2 | P_0 | P_2 |
| Parent 2: | | | | | | | | Offspring 2: | | | | | | | |
| T_1 | T_2 | T_4 | T_3 | T_5 | T_7 | T_8 | T_6 | T_1 | T_2 | T_4 | T_3 | T_5 | T_7 | T_8 | T_6 |
| P_1 | P_2 | P_0 | P_1 | P_2 | P_2 | P_0 | P_1 | P_1 | P_2 | P_0 | P_1 | P_1 | P_0 | P_0 | P_1 |

Fig. 6. An example of crossover

Reproduction also involves mutation, a random swap of processor assigned to two tasks in each selected chromosome of the new population. The primary purpose of mutation is to increase variation in a population.

3 Experimental Results

In this section the performance results and comparison of our proposed genetic algorithm is presented. In Fig. 7, our proposed genetic scheduling algorithm is compared with the well known algorithm of MCP [20] and five other scheduling algorithms [3], [5], [11], [12], [17]. The comparison is made by applying our algorithm to the task graph presented in reference [1] and used by MCP and the other five algorithms. The task graph of reference [2] is redrawn in Fig. 1. The results of comparison are shown in Fig. 7. It is observed that the proposed genetic algorithm results in a better scheduling than algorithms (3) to (7) and it produces the same result as LC. However, in LC algorithm the number of processors is determined by the algorithm [17]. In addition since scheduling is NP-hard, the proposed algorithm runs faster when the number of tasks grows.

- (a) The Proposed Genetic Algorithm (PGA) parameters:
 Population Size = 30, Number of Generations = 150, Number of Processors = 3,
 Crossover probability = 0.8, Mutation probability = 0.05.

(b) Scheduling results:

| Algorithm [Ref. No.] (Alg. No.) | Our PGA (1) | LC [17] (2) | EZ [12] (3) | HLFET [5] (4) | ETF [11] (5) | LAST [3] (6) | MCP [20] (7) |
|---------------------------------------|----------------|-------------------|-------------------|---------------------|--------------------|--------------------|--------------------|
| Completion Time | 39 | 39 | 40 | 41 | 41 | 43 | 40 |

(c) Scheduling produced by proposed GA (PGA):

| | | | | | | | | | | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|----------|----------|-------|----------|----------|----------|----------|----------|----------|
| T_1 | T_2 | T_3 | T_4 | T_6 | T_5 | T_8 | T_9 | T_{10} | T_{11} | T_7 | T_{12} | T_{14} | T_{13} | T_{15} | T_{17} | T_{16} |
| P_1 | P_2 | P_2 | P_1 | P_1 | P_2 | P_2 | P_2 | P_1 | P_0 | P_0 | P_0 | P_0 | P_1 | P_0 | P_0 | P_0 |

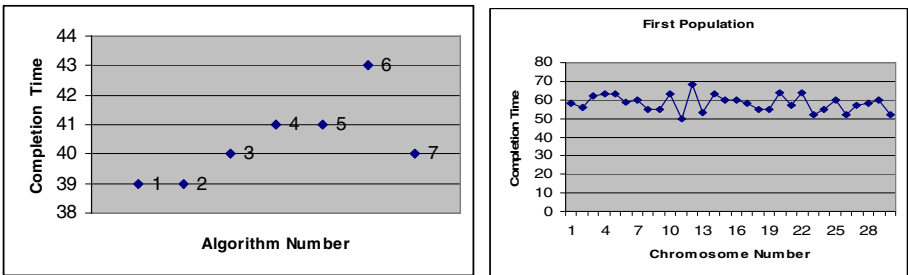


Fig. 7. A Comparison of the completion times for scheduling a task graph with 7 algorithms

The evolution of the chromosomes, while applying our proposed genetic algorithm in the above example, is shown below in Fig. 8. Also, in order to demonstrate the reliability of the result obtained in the above example, the results obtained by thirty runs of the algorithm to schedule the task graph [1] are compared in Fig. 8.

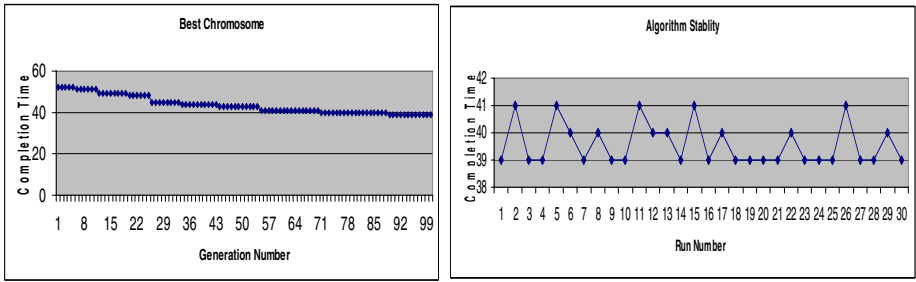
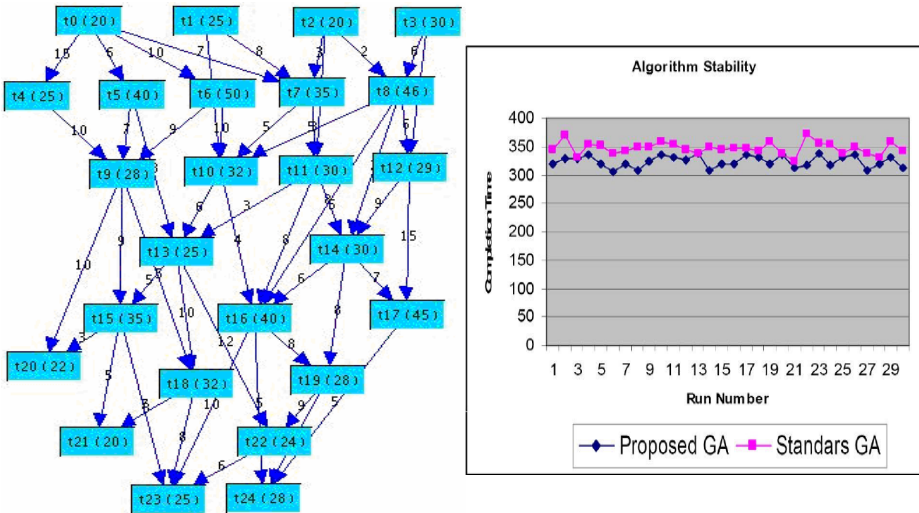


Fig. 8. Best fitness in 100 generations and fitness obtained in 30 runs

Below in Fig. 9, the results of comparing the proposed algorithm with an algorithm called Standard GA are shown. To demonstrate the stability of the results, both the algorithms have been applied 30 times to the random graph depicted is Fig. 9. Also in Fig. 9, the qualities of the first generations produced when applying the two algorithms are compared.

Genetic Algorithm parameters: Population Size = 30, Number of Generations = 150, Number of Processors = 3, Crossover Probability = 80%, Mutation Probability = 5%.



(a) Proposed GA., Fitness = 302

| | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0 | 1 | 3 | 5 | 2 | 6 | 8 | 4 | 7 | 9 | 11 | 12 | 10 | 13 | 14 | 15 | 16 | 17 | 20 | 18 | 21 | 19 | 24 | 22 | 23 |
| 0 | 1 | 2 | 0 | 1 | 0 | 2 | 0 | 1 | 0 | 1 | 2 | 1 | 0 | 2 | 0 | 1 | 2 | 2 | 0 | 2 | 1 | 2 | 1 | 1 |

(b) Standard GA., Fitness = 330

| | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|----|---|----|----|----|----|---|----|----|----|----|----|----|----|----|----|----|
| 3 | 1 | 0 | 5 | 2 | 8 | 7 | 4 | 11 | 6 | 10 | 12 | 14 | 13 | 9 | 17 | 16 | 18 | 15 | 21 | 20 | 19 | 22 | 23 | 24 |
| 2 | 2 | 1 | 0 | 0 | 2 | 0 | 1 | 0 | 1 | 0 | 2 | 2 | 0 | 1 | 2 | 1 | 1 | 0 | 2 | 1 | 2 | 0 | 1 | 2 |

Fig. 9. Comparison of the proposed and the Standard genetic scheduling algorithm

| GRAPH | Serial Time | Nodes | Edges | CGL | BCGA | PGA | Improvement % |
|-------|-------------|-------|-------|-----|------|-----|---------------|
| FFT1 | 296 | 28 | 32 | 152 | 124 | 124 | 0% |
| FFT2 | 760 | 28 | 32 | 270 | 240 | 195 | 19.75% |
| FFT4 | 480 | 28 | 32 | 260 | 255 | 185 | 27.4% |
| IRR | 1330 | 41 | 69 | 600 | 580 | 475 | 18.1% |

(a)

| GRAPH | Serial Time | CLANS | DSC | MCP | LC | LAST | PGA |
|-------|-------------|-----------|------------|-----------|-----------|-----------|--|
| FFT1 | 296/ 1 | 124/ 4 | 124/ 4 | 148/ 8 | 172/ 8 | 146/ 4 | 124/4 126/8 |
| FFT2 | 780/ 1 | 200/ 8 | 205/ 8 | 205/ 8 | 225/ 8 | 240/ 8 | 190/8 |
| FFT4 | 3440/ 1 | 405/ 4 | 710/ 12 | 710/ 8 | 710/ 8 | 170/ 8 | 185/4 685/8 685/ 12 |
| IRR | 1330/ 1 | 475 | 18.1% | 600 | 580 | 475 | 18.1% |

(b)

Fig. 10. Comparison of applying CGL, BCGA and PGA on FFT graphs (a) and Parallel time and number of processors (b)

Fast Fourier Transformation (FFT) and Internal Rate of Return IRR graphs provide a useful benchmark for evaluating the performance of scheduling algorithms [15]. In Fig. 10 (a), the results of applying our proposed algorithm, PGA, on FFT and IRR graphs are compared with the results presented in [15].

A comparative analysis of the speedup and performance can be delivered by comparing the results of applying the proposed algorithm, PGA, on the FFT and IRR graphs with the results in reference [4]. Speedup is defined as the ratio of serial time to parallel time and efficiency is the ratio of speedup to the number of processors used. In Fig. 10 (b) the parallel time and number of processors used, is shown.

4 Conclusions

Task graph scheduling problem is NP-hard. Scheduling can be considered as an optimization problem. To solve this NP-hard problem, non-deterministic approaches such as genetic algorithms are quite effective.

A task graph scheduling can be best encoded as an array, where each cell of the array represents a task and the processor to which the task is assigned. Considering the fact that the completion time of a scheduling is directly affected by the length of the critical path of the corresponding task graph, the objective of a genetic scheduling algorithm can be to minimize the length of the critical path. In this way, the fitness of each chromosome in a population is dependent on the length of its critical path.

References

1. Ahmad, I., Kwok, Y. K.: On Parallelizing the Multiprocessor Scheduling Problem. IEEE Transaction on Parallel and Distributed Systems, Vol. 10, No. 4 (1999) 414-432.
2. Al-Mouhamed, M. A: Lower Bound on the Number of Processors and Time for Scheduling Precedence Graphs with Communication Costs. IEEE Transaction Software Engineering, Vol. 16, No. 12 (1990) 1390-1401.

3. Baxter, J., Patel, J. H.: The LAST Algorithm: A Heuristic-Based Static Task Allocation Algorithm. *Proceeding of International Conference on Parallel Processing*, Vol. 2 (1989) 217-222.
4. Brest, J., Zumer, V.: A Comparison of the Static Task Graph Scheduling Algorithms. Faculty of Electrical Engineering and Computer Science Smetanova c. 17, Maribor, Slovenia (2000).
5. Coffman, E. G.: *Computer and Job-Shop Scheduling Theory*. John-Wiley (1976).
6. Correa, R. C., Ferreira, A., Rebreyend, P.: Scheduling Multiprocessor Tasks with Genetic Algorithms. *IEEE Transaction on Parallel and Distributed Systems*, Vol. 10, No. 8 (1999) 825-837.
7. De Jong, K. A., Spears, W. M.: A Formal analysis of the Role of Multi-Point Crossover in Genetic Algorithms. (1991).
8. Dhodhi, M. K., Ahmad, I.: Multiprocessor Scheduling Scheme Using Problem-Space Genetic Algorithms. *Proceeding of IEEE International Conference on Evolutionary Computation* (1995).
9. Goldberg, D. E.: *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley (1989).
10. Hou, E. S. H., Ansari, N., Ren, H.: A Genetic Algorithm for Multiprocessor Scheduling. *IEEE Transaction on Parallel and Distributed Computing*, Vol. 5, No. 2 (1994) 113-120.
11. Hwang, J. J., Chow, Y. C., Anger, F. D., Lee, C. Y.: Scheduling Precedence Graphs in Systems with Inter-processor Communication Times. *SIAM Journal on Computer*, Vol. 18, No. 2 (1989) 244-257.
12. Kim, S. J. and Browne, J. C.: A General Approach to Mapping of Parallel Computation upon Multiprocessor Architectures. *Proceeding Of International Conference on Parallel Processing*, Vol. 2 (1988) 1-8.
13. Kruatrachue, B., Lewis, T. G.: Duplication Scheduling Heuristics (DSH): A New Precedence Task Scheduler for Parallel Processor Systems. Technical Report. Oregon State University, Corvallis (1987).
14. Kwok, Y. K., Ahmad, I.: Benchmarking and Comparison of the Task Graph Scheduling Algorithms. *Journal of Parallel and Distributed Computing* 59 (1999) 381-422.
15. McCreary, C. L., Khan, A. A., Thompson, J. J., McArdle, M. E.: A Comparison of Heuristics for Scheduling DAGS on Multiprocessors. *Proceedings of the 8th International Parallel Processing Symposium* (1994) 446-451.
16. Rinehart, M., Kianzad, V., Bhattacharyya, Sh. S.: A Modular Genetic Algorithm for Scheduling Task Graphs. Department of Electrical and Computer Engineering, and Institute for Advanced Computer Studies, University of Maryland, College Park (2003).
17. Sarkar, V.: *Partitioning and Scheduling Parallel Programs for Multiprocessors*. MIT Press, Cambridge, MA (1989).
18. Sih, G. C., Lee, E. A.: A Compile-Time Scheduling Heuristic for Interconnection Constrained Heterogeneous Processor Architectures. *IEEE Transaction on Parallel and Distributed Systems*, Vol. 4, No. 2 (1993) 75-87.
19. Wu, A. S., Yu, H., Jin, Sh., Lin, K. Ch., Schiavone, G.: An Incremental Genetic Algorithm Approach to Multiprocessor Scheduling. *IEEE Transaction on Parallel and Distributed Systems*, Vol. 15, No. 9 (2004) 824-834.
20. Wu, M. Y.: MCP Revisited. Department of Electrical and Computer Engineering, University of New Mexico.

Universal Quantum Gates Via Yang-Baxterization of Dihedral Quantum Double

Mario Vélez* and Juan Ospina

Departamento de Ciencias Básicas, Universidad EAFIT, A.A. 3300
Medellín, Colombia
{mvelez,jospina}@eafit.edu.co

Abstract. The recently discovered Yang-Baxterization process for the quantum double of the dihedral group algebra, is presented keeping on mind the quantum computation. The products resultant from Yang-Baxterization process are interpreted as universal quantum gates using the Brylinski's theorem. Results are obtained for two-qubits and two-qutrits gates. Using the Zhang-Kauffman-Ge method (ZKGM), certain Hamiltonians responsible for the quantum evolution of the quantum gates are obtained. Possible physical systems such as anyons systems are mentioned as referents for practical implementation.

1 Introduction

In the present work we intend to use the methodology given in [1] with the aim to exploit within the quantum computing the findings of reference [2]. This methodology consists in to construct the quantum gates, starting from unitary representations of certain groups and algebras which appear in geometric topology and they refer direct and naturally to structures of topological entanglement. Examples of such algebraic structure are: the braid group, the Iwahori-Hecke algebra [3], the Temperley-Lieb algebra [4], the Birman-Wenzl-Murakami algebra [5], and so on. Examples of structures with topological entanglement are braids, knots, links, tangles, and so on.

From the other side, a very similar equations to the braid relation, appear in physics as the integrability conditions for certain systems of statistical physics [6]. Such equations are generically named the Yang-Baxter equations (YBE). One of this equations is called the QYBE and carries certain spectral parameter. Recently has been proposed that universal quantum gates can be derived from unitary solutions of the QYBE [1], being such unitary solutions derived, using a procedure which is named Yang-Baxterization [7]. Also recently the Yang-Baxterization of the quantum doubles of finite groups such as the dihedral group [2] was discovered and associated with certain kinds of anyon systems. In the present work we intend to use the methodology given in [1] with the aim to exploit within the quantum computing the findings of reference [2]. Our principal objective here is to understand more clearly the relationships among quantum universality, quantum entanglement, topological entanglement and anyon computation.

* This research was supported by EAFIT UNIVERSITY (grant # PY0511).

2 The Zhang-Kauffman-Ge Method

In this section we collect the basic elements to understand the ZKGM of Yang-Baxterization that produces universal quantum gates and also possibly quantum algorithms for the computation of the knot invariants. (In the present work only are considered the questions about the relation between Yang-Baxterization and universal quantum gates via the Bryliski theorem [1]. The exploration of the relations between Yang-Baxterization and knot invariants via quantum algorithms will be realized in future investigations). The basic elements of the ZKGM are: the braided YBE and the QYBE. The braided YBE is defined as

$$(\check{R} \otimes \mathbb{1})(\mathbb{1} \otimes \check{R})(\check{R} \otimes \mathbb{1}) = (\mathbb{1} \otimes \check{R})(\check{R} \otimes \mathbb{1})(\mathbb{1} \otimes \check{R}), \quad (1)$$

and the QYBE has the form

$$(\check{R}(x) \otimes \mathbb{1})(\mathbb{1} \otimes \check{R}(xy))(\check{R}(y) \otimes \mathbb{1}) = (\mathbb{1} \otimes \check{R}(y))(\check{R}(xy) \otimes \mathbb{1})(\mathbb{1} \otimes \check{R}(x)), \quad (2)$$

where \check{R} and $\check{R}(x)$ are square matrices of order d^2 and $\mathbb{1}$ is the identity matrix of order d , being d the dimension of certain Hilbert space. With these elements, it is possible to understand the Yang-Baxterization procedure as one that derives a unitary solution of the QYBE (2) starting from a unitary solution of the braided YBE (1). Note that this procedure is originally associated with the braided YBE directly derived from the braid relation of the braid group.

The solutions of (1) and (2) are square matrices of dimensions 4, 9, 16, The most simple solutions appear obviously for dimensions 4, 9. The solutions with four dimensions correspond with universal quantum gates for a system of two qubits. The solutions with nine dimensions corresponds with universal quantum gates for a system of two qutrits. In the present work only the cases for qubits and qutrits are considered. Now the ZKGM proposes to use the brylinskis theorem with the purpose to obtain the range of variation of the spectral parameter x , for which the solution of (2) is a universal quantum gate. Finally the ZKGM introduces an evolution equation of the Schrödinger kind which has the form

$$i \frac{\partial \psi(x)}{\partial x} = H(x) \psi(x), \quad H(x) = i \frac{\partial \check{R}(x)}{\partial x} \check{R}^{-1}(x). \quad (3)$$

In the equation (3), the spectral parameter x takes the role of the time t . When the equation (3) is applied for a matrix that is solution of (2) a hamiltonian is obtained. This hamiltonian is certain matrix which can be rewritten as a sum of tensorial products on Pauli matrices. For hence it is possible to consider the resultant hamiltonian like one that corresponds to a system of spin 1/2 chains. This physical system is the referent for certain kind of topological quantum computation. More details can be found on [1]. The principal purpose of the ZKGM is to understand the complex relations among quantum universality, topological entanglement and quantum entanglement.

3 The Dancer-Isaac-Links Method

The ZKGM entails the application of the Yang-Baxterization of the braid group. It is clear that is possible to apply the ZKGM for other algebras that appear in geometric topology such as previously mentioned, Temperley-Lieb, Birman-Wenzl-Murakami, and their generalizations. In the present work we intend to apply in quantum computing the Dancer-Isaac-Links method (DILM) of Yang-Baxterization which uses the dihedral group or more technically the quantum double of the group algebra of the dihedral group $D(D_n)$. The Yang-Baxterization procedure used by the DILM is the same that is used in the ZKGM. The principal difference between DILM and ZKGM is that ZKGM applies the representations of the braid group B_n while the DILM applies the representations of $D(D_n)$. Other difference is that the braid group B_n of the ZKGM is directly related to the topological entanglement while for the $D(D_n)$ of the DILM it is not the case. But these methods have in common that the corresponding algebras can be considered as fusion and braiding rules for systems of anyons.

Finally within the DILM is possible to construct universal quantum gates and hamiltonians, but such constructions are not presented in the fundamental reference [2]. Our principal contributions in this work will be presented in the following two sections on which we use the results of DILM according with the lines on quantum computing proposed in the ZKGM.

4 Two-Qubits Universal Quantum Gates

We start here from the matrix taken from [2]

$$\check{R} = \begin{bmatrix} \omega^j & 0 & 0 & 0 \\ 0 & 0 & \omega^{-j} & 0 \\ 0 & \omega^{-j} & 0 & 0 \\ 0 & 0 & 0 & \omega^j \end{bmatrix}, \quad (4)$$

where $\omega = \exp(2\pi i/n)$ and $0 \leq j < n$. The \check{R} matrix given by (4) is a unitary matrix and it is a solution of the braided YBE. Such matrix is naturally associated with the two-dimensional irreps of $D(D_n)$ [2]. We consider the matrix \check{R} of (4), being unitary, as a universal quantum gate and given that \check{R} of (4) is a representation of $D(D_n)$ which is the dynamical algebra of a certain system of anyons, then the \check{R} can be considered as an operator that performs anyonic entanglement, it is to say an operator that performs fusion and braiding for anyons. >From other side, it is possible to show that \check{R} of (4) also perform quantum entanglement when such matrix actuates on Hilbert spaces that represent quantum states of two qubits. The action of \check{R} over the basic states of two qubits is

$$\begin{aligned} \check{R} |00\rangle &= \omega^j |00\rangle, \quad \check{R} |01\rangle = \omega^{-j} |10\rangle, \\ \check{R} |10\rangle &= \omega^{-j} |01\rangle, \quad \check{R} |11\rangle = \omega^j |11\rangle. \end{aligned} \quad (5)$$

At the sequel we consider now the following quantum state for a single qubit $|\psi\rangle = |0\rangle + |1\rangle$, and for hence we can obtain the following disentangled quantum state for two qubits

$$|\psi\rangle \otimes |\psi\rangle = |00\rangle + |01\rangle + |10\rangle + |11\rangle . \tag{6}$$

Now the action of the quantum gate \check{R} over the disentangled state (6) gives

$$|\phi\rangle = \check{R}(|\psi\rangle \otimes |\psi\rangle) = w^j |00\rangle + w^{-j} |01\rangle + w^{-j} |10\rangle + w^j |11\rangle . \tag{7}$$

The resultant state (7) is entangled when $w^{2j} \neq w^{-2j}$, or equivalently when $j \neq 0, j \neq n/4, j \neq n/2, j \neq 3n/4, j \neq n$. Now from Yang-Baxterization the following $\check{R}(x)$ matrix is obtained [2]

$$\check{R}(x) = \begin{bmatrix} \omega^j - \omega^{-3j}x^2 & 0 & 0 & 0 \\ 0 & (\omega^j - \omega^{-3j})x & -\omega^{-j}(x^2 - 1) & 0 \\ 0 & -\omega^{-j}(x^2 - 1) & (\omega^j - \omega^{-3j})x & 0 \\ 0 & 0 & 0 & \omega^j - \omega^{-3j}x^2 \end{bmatrix} . \tag{8}$$

The matrix of (8) satisfies the following unitary condition [2]

$$\check{R}(x)\check{R}(x^{-1}) = (w^{-6j} + w^{2j} - \frac{1}{w^{2j}x^2} - \frac{x^2}{w^{2j}}) \mathbb{I}_2 \otimes \mathbb{I}_2 . \tag{9}$$

In the reference [8] the unitary braiding operators were considered as universal quantum gates. Posteriorly, in the reference [11], the unitary $\check{R}(x)$ -matrices were viewed as universal quantum gates with the help of the Brylinski's theorem. Now in this subsection we apply the ideas of [11] for the case of the $\check{R}(x)$ -matrix given by (8). The equation (10) shows the action of the operator $\check{R}(x)$ of (8) over an initial state denoted V_1

$$V_1 = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} , \quad V_2 = R(x) V_1 = \begin{bmatrix} (\omega^j - \omega^{-3j}x^2) a \\ (\omega^j - \omega^{-3j})xb - \omega^{-j}(x^2 - 1)c \\ -\omega^{-j}(x^2 - 1)b + (\omega^j - \omega^{-3j})xc \\ (\omega^j - \omega^{-3j}x^2) d \end{bmatrix} , \tag{10}$$

where V_2 represents the final state. Now according with the Brylinski theorem, the matrix of (8) is universal if and only if the following entanglement condition is satisfied

$$K_1^2(x)ad - bc(K_2^2(x) + K_3^2(x)) - K_2(x)K_3(x)(b^2 + c^2) \neq 0 . \tag{11}$$

Assuming that the initial state V_1 is disentangled ($ad = bc$), then the universality condition is changed to

$$K_1^2(x)bc - bc(K_2^2(x) + K_3^2(x)) - K_2(x)K_3(x)(b^2 + c^2) \neq 0 . \tag{12}$$

where

$$K_1(x) = \omega^j - \omega^{-3j}x^2, K_2(x) = (\omega^j - \omega^{-3j})x, K_3(x) = -\omega^{-j}(x^2 - 1). \quad (13)$$

Using the formula for the hamiltonian given by (3) and using the matrix of (8) we obtain the following hamiltonian

$$H(x) = \begin{bmatrix} F_1(x) & 0 & 0 & 0 \\ 0 & F_2(x) & F_3(x) & 0 \\ 0 & F_3(x) & F_2(x) & 0 \\ 0 & 0 & 0 & F_1(x) \end{bmatrix}, \quad (14)$$

where

$$F_1(x) = \frac{-iB_1(x)\sqrt{\rho(x)}}{-\omega^j + \omega^{-3j}x^2}, \quad (15)$$

$$F_2(x) = \frac{-i\sqrt{\rho(x)}(B_2(x)\omega^{5j}(x^2 - 1) + B_3(x)\omega^{3j}x(\omega^{4j} - 1))}{-x^2\omega^{8j} - x^2 + x^4\omega^{4j} + \omega^{4j}}, \quad (16)$$

$$F_3(x) = \frac{-i\sqrt{\rho(x)}(B_2(x)\omega^{3j}x(\omega^{4j} - 1) + B_3(x)\omega^{5j}(x^2 - 1))}{-x^2\omega^{8j} - x^2 + x^4\omega^{4j} + \omega^{4j}}, \quad (17)$$

and being

$$B_1(x) = -2 \frac{\omega^{-3j}x}{\sqrt{\rho(x)}} - \frac{1}{2} \frac{(\omega^j - \omega^{-3j}x^2) \frac{d}{dx}\rho(x)}{(\rho(x))^{3/2}}, \quad (18)$$

$$B_2(x) = -2 \frac{\omega^{-j}x}{\sqrt{\rho(x)}} + \frac{1}{2} \frac{\omega^{-j}(x^2 - 1) \frac{d}{dx}\rho(x)}{(\rho(x))^{3/2}}, \quad (19)$$

$$B_3(x) = \frac{\omega^j - \omega^{-3j}}{\sqrt{\rho(x)}} - \frac{1}{2} \frac{(\omega^j - \omega^{-3j})x \frac{d}{dx}\rho(x)}{(\rho(x))^{3/2}}. \quad (20)$$

The hamiltonian (14) can be rewritten as

$$H(x) = \frac{1}{2}F_1(x)(\sigma_z \otimes \sigma_z + I \otimes I) - \frac{1}{4}F_2(x)(\sigma_z + I) \otimes (\sigma_z - I) - \frac{1}{4}F_2(x)(\sigma_z - I) \otimes (\sigma_z + I) + F_3(x)(\sigma_1 \otimes \sigma_2 + \sigma_2 \otimes \sigma_1), \quad (21)$$

where we use the operators $\sigma_x, \sigma_y, \sigma_z, \sigma_1, \sigma_2$ for a particles of spin 1/2.

We show that the DILM when is applied according with the lines of ZKGM, produces universal quantum gates via yang-baxterization for various values of the spectral parameter. The $\hat{R}(x)$ given by (8) is able to generate quantum entanglement but it has no direct relationship with the topological entanglement. This matrix has an origin inside anyon systems with quantum dihedral symmetry(dihedral anyons). These indicate that results that were presented in this section may useful for to understand the relations between quantum universality, quantum entanglement, topological entanglement and dihedral anyons.

5 Two-Qutrits Universal Quantum Gates

In quantum computing may be necessary the introduction of one and two- qutrits universal quantum gates. For then, in this section we show the construction of two-qutrits universal quantum gates via Yang-Baxterization of three dimensional representations of the quantum double of certain dihedral group algebras. The procedure is similar that it was presented in the previous section and for hence we just will show the final results. In the reference [2] was discovered a new 21-vertex model for a system of anyons with $D(D_3)$ or $D(D_6)$ symmetry with a fundamental representation which is three-dimensional. We argue here that such system of anyons can be used for implementation and processing of systems with one and two qutrits. Particularly via Yang-Baxterization we obtain two-qutrits quantum gates. The details are as follows. We start from the \check{R} given by [2]

$$\check{R} = (-1)^b \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & (-1)^a & 0 \\ 0 & 0 & (-1)^a & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & (-1)^a & 0 & 0 \\ 0 & (-1)^a & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad (22)$$

with $a = b = 0$, whose eigenvalues are:

$$\{(-1)^{\frac{2}{3}}, (-1)^{\frac{2}{3}}, -(-1)^{\frac{1}{3}}, -(-1)^{\frac{1}{3}}, 1, 1, 1, 1, 1\}. \quad (23)$$

Via Yang-Baxterization the following $\check{R}(x)$ matrix is obtained [2]

$$\check{R}(x) = \begin{bmatrix} G_1(x) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & G_2(x) & 0 & 0 & 0 & G_3(x) & G_4(x) & 0 & 0 \\ 0 & 0 & G_2(x) & G_4(x) & 0 & 0 & 0 & G_3(x) & 0 \\ 0 & 0 & G_3(x) & G_2(x) & 0 & 0 & 0 & G_4(x) & 0 \\ 0 & 0 & 0 & 0 & G_1(x) & 0 & 0 & 0 & 0 \\ 0 & G_4(x) & 0 & 0 & 0 & G_2(x) & G_3(x) & 0 & 0 \\ 0 & G_3(x) & 0 & 0 & 0 & G_4(x) & G_2(x) & 0 & 0 \\ 0 & 0 & G_4(x) & G_3(x) & 0 & 0 & 0 & G_2(x) & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & G_1(x) \end{bmatrix}, \quad (24)$$

were $G_1(x) = x^2 - x + 1$, $G_2(x) = x$, $G_3(x) = 1 - x$, $G_4(x) = x(x - 1)$. The matrix of (24) satisfies the following unitary condition [2]

$$\check{R}(x)\check{R}(x^{-1}) = \left[\frac{G_1(x)}{G_2(x)} \right]^2 \mathbb{I}_3 \otimes \mathbb{I}_3, \quad (25)$$

which is immediately verified. Finally from the $\check{R}(x)$ matrix of (24), the following hamiltonian is derived

$$H(x) = \begin{bmatrix} F_1(x) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & F_2(x) & 0 & 0 & 0 & F_3(x) & F_4(x) & 0 & 0 \\ 0 & 0 & F_2(x) & F_4(x) & 0 & 0 & 0 & F_3(x) & 0 \\ 0 & 0 & F_3(x) & F_2(x) & 0 & 0 & 0 & F_4(x) & 0 \\ 0 & 0 & 0 & 0 & F_1(x) & 0 & 0 & 0 & 0 \\ 0 & F_4(x) & 0 & 0 & 0 & F_2(x) & F_3(x) & 0 & 0 \\ 0 & F_3(x) & 0 & 0 & 0 & F_4(x) & F_2(x) & 0 & 0 \\ 0 & 0 & F_4(x) & F_3(x) & 0 & 0 & 0 & F_2(x) & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & F_1(x) \end{bmatrix}, \quad (26)$$

where

$$F_1(x) = \frac{iA_1(x)\sqrt{\rho(x)}}{G_1(x)}, \quad (27)$$

$$F_2(x) = \frac{-i\sqrt{\rho(x)}(A_4(x)G_4(x) - A_3(x)G_3(x) - A_2(x)G_2(x))}{G_1^2(x)}, \quad (28)$$

$$F_3(x) = \frac{i\sqrt{\rho(x)}(A_2(x)G_4(x) + A_4(x)G_3(x) + A_3(x)G_2(x))}{G_1^2(x)}, \quad (29)$$

$$F_4(x) = \frac{i\sqrt{\rho(x)}(A_3(x)G_4(x) + A_2(x)G_3(x) + A_4(x)G_2(x))}{G_1^2(x)}, \quad (30)$$

being

$$A_i(x) = \frac{\frac{d}{dx}G_i(x)}{\sqrt{\rho(x)}} - \frac{1}{2} \frac{G_i(x) \frac{d}{dx}\rho(x)}{(\rho(x))^{3/2}} \quad \text{for } i = 1, 2, 3, 4. \quad (31)$$

The hamiltonian (26) is a nine-dimensional operator which can be rewritten as a sum of tensorial products between three-dimensional spin-1 operators. This hamiltonian corresponds to a chain of spin-1 anyons and such system is able to process qutrits. It worthwhile to remark that the $\check{R}(x)$ given by (24) has no an intrinsic topological origin, since the dihedral group is very different to the braid group, but this matrix is able to produce quantum entanglement despite it has no direct relation with topological entanglement. This matrix has a direct relation with dihedral anyon systems and for hence its universality is a consequence of the anyon physics. The systems of qutrits are expected have an amplified power of computation with respect to the system of qubits. The results thrown some light about the relations among quantum entanglement, topological entanglement, universality and anyon computation, from a more wide perspective that it corresponds to the standard qubits.

6 Conclusions

Our principal contribution in this paper to the anyonic topological quantum computing was the application of DILM in quantum computing with the guidance of the ZKGM. We believe that such application is new in quantum computation. The principal novelty was the introduction of dihedral anyons without the necessity to invoke some kind of structure of topological entanglement (braids). We use only the dihedral quantum double when this is assumed as the dynamical symmetry for certain class of anyon systems. In this way the universal quantum gates have a physical referent from the beginning which it is not the case for the ZKGM.

With respect to the future lines of research is worthwhile to note here that ZKGM can be extended in such way that is possible to have quantum algorithms for the computation of invariants for knots and links, starting from the representations of braid group. In this perspective is very interesting to consider the possibility of existence of quantum algorithms for link invariants starting from representations of $D(D_n)$.

Acknowledgements. We thank to the Professor Félix Londoño, director of research and teaching office in EAFIT University, for permanent support and orientation. We thank to the Professor Andrés Sicard Ramírez, director of the Logic and Computation Group of EAFIT, for constant support, feedback and guidance.

References

1. Zhang, Y., Kauffman, L.H., Ge, M.L.: Yang–baxterizations, universal quantum gates and hamiltonians. *Quant. Inf. Proc.* **4** (2005) 159–197
2. Dancer, K., Isaac, P., Links, J.: Representations of the quantum doubles of finite group algebras and spectral parameter dependent solutions of the yang-baxter equation. Eprint: [arXiv.org/abs/math.QA/0511072v2](https://arxiv.org/abs/math.QA/0511072v2) (2006)
3. Bigelow, S.: Braid groups and iwahori-hecke algebras. Preprint: [arXiv.org/abs/math.GT/0505064](https://arxiv.org/abs/math.GT/0505064) (2005)
4. Temperley, H., Lieb, E.: Relations between the ‘percolation’ and ‘colouring’ problem and other graph-theoretical problems associated with regular planar lattices: Some exact results for the ‘percolation’ problem. *Proceedings of the Royal Society of London A* **32** (1971) 251–280
5. Birman, J., Wenzl, H.: Braids, link polynomials and a new algebra. *Transactions of the American Mathematical Society* **313** (1989) 249–273
6. Baxter, R.: Exactly solved models in statistical mechanics. Academic Press (1982)
7. Jones, V.: Baxterization. *Int. J. Mod. Phys. B* **4** (1990) 701–714
8. Kauffman, L., Jr., S.L.: Brading operators are universal quantum gates. *New J. Phys.* **6** (2004) Preprint: [arXiv.org/abs/quant-ph/0401090](https://arxiv.org/abs/quant-ph/0401090).

Evolutionary Bi-objective Learning with Lowest Complexity in Neural Networks: Empirical Comparisons

Yamina Mohamed Ben Ali

Research Group on Artificial Intelligence, Computer Science Department,
Badji Mokhtar University BP 12, Annaba, Algeria
benaliyam2@yahoo.fr

Abstract. This paper introduces a new study in evolutionary computation technique in order to learn optimal configuration of a multilayer neural network. Inspired from thermodynamic perception, the used evolutionary framework undertakes the optimal configuration problem as a Bi-objective optimization problem. The first objective aims to learn optimal layer topology by considering optimal nodes and optimal connections by nodes. Second objective aims to learn optimal weights setting. The evaluation function of both concurrent objectives is founded on an entropy function which leads the global system to optimal generalization point. Thus, the evolutionary framework shows salient improvements in both modeling and results. The performance of the required algorithms was compared to estimations distribution algorithms in addition to the Backpropagation training algorithm.

1 Introduction

Designing the architecture and correct parameters for the learning algorithm is a tedious task for modeling an optimal artificial neural network with better generalization performance. Often, the emerging needs to hybrid intelligent systems overcomes neural networks limits and becomes an optimal way to achieve synergic neural systems [2], [3], [18]. Despite the fact that many intelligent systems undertake to increase neural performance, the co-evolution of learning and self-organization behaviors become the best natural means to highlight the fundamental behavior criterion of neural adaptation. Hence, one of the most commonly ascribed approaches to reach adaptive continuous updating consisted in matching the neural network modeling capabilities with the adaptation properties of evolutionary computation [4], [5], [8].

Evolutionary Artificial Neural Networks EANN framework conjugates the potentialities of evolutionary computation and the neural networks capabilities [4], [9], [18], [12] to optimize several tasks like connection weights training, architecture topologies [1], [10], [11], [19], [20], [21], [22]. Although EANN formalism raised and many satisfactory results obtained, the subject receives still and much frequently interest. Evolving neural system to acquire concurrently optimal neural topology and optimal weights is always an attractive goal, especially when it is always possible to improve generalization performance. Moreover, to draw such perspective, our research studies were motivated by the following observations:

- Demonstrate the strongly reliability of a neural network on evolutionary computation. The causality relation implies the dominance of the evolution metaphor which ensures the training of a neural network. Herein, evolutionary computation largely contributes to outperform neural learning than the opposite case.
- Increase the effectiveness of both the accuracy of classification and generalization. This is meaningful by integrating self-adaptive behavior as a relevant characteristic to guide the evolution of optimal neural network configuration.
- Map a new semantic to the evolution mechanic by including a thermodynamic evolution principal as evaluation function of the global system. Also, it is worthwhile to note that the process of problem solving develops an evolutionary dynamism which considerably raises the system capacities through a lower order convergence time complexity.

In the following, we introduce Sect. 2 that exposes briefly some notions and notations about multilayer neural network. Section 3 explains the proposed framework for evolubility of a neural network; Sect. 4 describes experiments for classification problems which lead to some comparisons between estimation distribution algorithms and Backpropagation algorithm. Finally, Sect. 5 concludes the elaborated working.

2 Artificial Neural Networks

In literature, a neural net is defined as a graph $G(N, A, \Psi)$, where N is a set of neurons (or nodes), A denotes neurons connections (also called arcs or synapses), and Ψ represents the leaning rule whereby are able to adjust the strengths of their connections. A neuron receives its inputs from an external source or from others neurons in the graph. It then undertakes some processing on this input and sends the result as output. The underlying function of a neuron is called the activation function. Hence, each neuron computes its activation as a weighted sum of the inputs to the node in addition to a constant value called the bias. From herein, the following notations will be used form a one hidden multilayer neural net:

- I and H are the number of input and hidden units respectively.
- (x^p, y^p) is the training pattern $x^p \in X = (x_1^p, x_2^p, \dots, x_I^p)$ and $p = 1 \dots P$, is p^{th} pattern in the input feature space X of dimension I , P is the total number of patterns, and $y^p \in Y$ the corresponding scalar of pattern x^p in the hypothesis space Y .
- w_{ih} and w_{ho} are the weights connecting input unit i to hidden unit h , and hidden unit h to output unit o respectively.
- $\Phi_h(x^p) = \sigma(a_h)$, $a_h = \sum_{i=1}^I w_{ih} w_i^p$, $h = 1 \dots H$, is the h^{th} hidden unit's output

corresponding to the input pattern x^p , where a_h is the activation function. The common node function (sigmoidal or logistic, tangent hyperbolic function etc) are differentiable to arbitrary order, which implies that the error is also differentiable to arbitrary order.

- $\hat{y}^p = \sigma(a_0)$, $a_0 = \sum_h w_{h0} \Phi_h(x^p)$ is the network output and a_0 is the activation of output unit o corresponding to the input pattern x^p .

In practice, multilayer neural networks are used as non-parametric regression methods that approximate underlying functionality in data by minimizing an error function. The notion of closeness on the training set P is typically formalized through an approximated empirically function defined as an error function such as $\sum_{p=1}^P (y^p - \hat{y}^p)^2$.

Obviously, in learning state the neural network requires more than one objective function. In addition to the error function which purpose is to evaluate the performance of the learning parameters, it is also important to specify the leaning rule that determines efficiently optimal weights. Commonly, the learning by iterative gradient descent such as Backpropagation and its other extensions seems to be adequate algorithms. But, in most cases these algorithms encounter certain difficulties in paractice, like: the convergence tends to be extremely slow, convergence to the global minimum is not guarantee, learning constants must be guessed heuristically.

3 Bi-objective Optimization Through Multi-scaled Learning

From the point of view of optimization, evolutionary computation is a powerful stochastic zeroth order method, requiring only values of the function to optimize. In literature, the reasons to choose evolutionary computation to deal with architecture design were summarized according to the following landscape criteria:

- Infinitely large surface
- Non-differentiable surface
- Complex and noisy surface (between architecture and performance)
- Deceptive surface (similar architecture, but different performance)
- Multimodal surface (different architecture, but similar performance)

In our current research studies, the evolutionary “operational” semantic is provided by a dynamism degree of evolution function. Indeed, this macroscopic landscape function which entails a pseudo-temperature T will be able to simulate the energy of the global system in order to drive the system to an equilibrium state with optimal configuration. Consequently, all neural elements become subject to microscopic laws of mechanics substantially guided by the dynamic of mutation rates.

The elaborated framework represents the first machine inspired by both statistical mechanics to learn mutation rates in order to update the neural environment structures (chromosomes), and thermodynamic metaphor as an evaluation heuristic to reach equilibrium state of the system. In other words, this machine is able to achieve the optimal convergence state relating on the maximum energy of the system at final state.

This new version of energy landscape *entropy-based* comes to improve some weakness of the system when we elaborated the first energy function which involved an exponential evaluating [14] and regardless to substantial mechanics.

Let us now drawing the global behavior of the system: a set of neural structures (connection chromosomes and weights chromosomes) are evolutionary evolved in time to reach optimal structures according to a learning kernel. This evolutionary multi-scaled kernel embodies an entropy function to both control the system equilibrium (maximum energy) and self-adapt microscopic laws of evolution probabilities. We expose at first scale the optimization mechanism of connection structures and in second scale the driven mechanism entropy-based which supervises both connection and weight structures.

3.1 Self-organization of Neural Maps

Self-organization of neural maps represents a learning task which aims to obtain dynamic and variable neural structures. The lowest network complexity is itself a bi-objective problem insofar as the learning procedure should be able to determine both the optimal layer topology (optimal hidden nodes) and the optimal fan-in (the reduced number of connections neighbors) for each hidden node. It is well known that a network that is too big for a particular classification task is more likely to overfit the training data and have poor performance on unseen examples (i.e., poor generalization) than a small network.

First Phase. We lunch the evolution according to a descendent learning approach in order to reduce the vicinity of the network (hidden nodes) for each network layer and for each neural network $\eta^i \in \Omega$ (Ω is the neural networks population). The descendent learning procedure applies mainly three steps:

- Configure the static design of an individual neural net η^i with h_{\max} hidden nodes in j^{th} hidden layer. The configuration number h_{\max} could be fixed either empirically or heuristically
- Construct for each hidden node k a randomly connection structure (by 0 or 1)
- Resize the connectivity of each layer according to $\text{SizeReduce}(\eta_j^i)$ function which applies two updating rules respectively defined by (1) and (2) :

$$\text{Connect}(\eta_{jk}^i) = \sum_{m \leq h_{j-1}} \varpi_m \quad (1)$$

It is assumed that if one connection link does not exist, it is automatically removed from the network.

$$\varpi_m = \begin{cases} 1 & \text{Connect}(\eta_{jk}^i) > 0 \\ 0 & \text{Connect}(\eta_{jk}^i) = 0 \end{cases} \quad (2)$$

Function SizeReduce(η_j^i)

Input: h_j

$h'_j \leftarrow h_j$ {Assuming that $h_j = h_{\max}$ at the beginning}

For each node $k=1$ to h_{j-1} then

If Connect(η_{jk}^i) = 0 then $h'_j = h'_j - 1$ { k^{th} hidden node is removed}

Compute Connect(η_{jk}^i)

Output h'_j

End.

Second Phase. This phase selects valid networks those considered as competitive in the population by means of a score evaluation function. The used heuristic relies on considering the state of hidden neurons. A hidden neuron is actually active when it has at least one connection link with neurons of lowest layer. The score of a neural η^i strongly depends on the score of each hidden layer net designed by ϖ_j , where $1 \leq j \leq N_H$ and N_H is the number of hidden layers. The schedule of the evaluation is described by the following main steps:

- Evaluate the score of η^i following learning heuristics described by (3) and (4) :

$$\text{Score}(\eta^i) = \sum_{j \leq N_H} \varpi_j \quad (3)$$

$$\varpi_j = \begin{cases} 1 & \text{if } act_j \geq \pi_j \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

act_j is the number of active neurons in layer j , and $\pi_j = \frac{H_j}{\alpha}$ is an empirical value with $\alpha = \{2, 3, 4\}$.

- If $\text{Score}(\eta^i) = N_H$ then we select η^i as a competitive solution; otherwise η^i will be rejected.

This evaluation is carried out each generation, and the net could eventually change its morphological aspect following the connection constraint as shown in Fig. 1. However, this method stills a best way to select optimal network configuration before to lunch the learning behavior to train weight neurons.

3.2 Thermodynamic Learning Entropy-Based

Inspired from both thermodynamic principles and statistical principles, the evolutionary learning kernel (ELK) embodies mainly an entropy function to measure the energy of the system in order to:

- Self-adapt stochastic evolution operators like mutation probabilities
- Self-adapt the selection rate at each generation
- Control obviously the equilibrium state of the system to stop evolution.

But as opposite to HelmHoltz free energy and Boltzmann machine, the principle of the system stability is inversed in the sense where the higher energy implies lower evolution probability and then a smooth stopping of the evolution.

To define the entropy in the genetic context, we must understand first the evolution principle of all individuals in the population. For this intention, we should firstly consider the environment energy Z as the difference ratio between the temperature T of the system at instant t and the maximal temperature T_{max} , secondly express the genetic entropy (5) as the phenotypic variability defined in terms of the environment energy. Thus, the entropy should be described by the following statement:

$$E = -a \text{Log}(Z) \tag{5}$$

Where $Z = \frac{T}{T_{max}}$, and $a = T^{-1}$ (all parameters vary in time).

At this stage, the entropy function acts as an adaptation function in a “chaotic” distribution. All following genotypic updating are strongly subject to the intensity of mutation probabilities described by equations (6) and (7):

$$p_i = \exp(-\sigma)^{\text{log}(t+1)} \tag{6}$$

$$p_i = -\text{Log}(E) \tag{7}$$

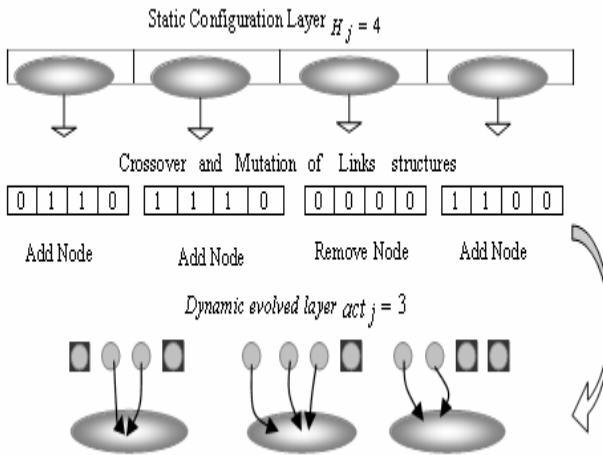


Fig. 1. Dynamic configuration of an optimal neural network

The parameterization of evolution speed up and the speed moving in a multi-dimensional search space constitutes our contribution to avoid extensive computational time.

```

Algorithm. Evolutionary neural learning entropy-based
   $t \leftarrow 0$  {counter generation}
  Initialize  $\Omega_s$  a random population with static nets
  Evaluate score-based  $\Omega_s$ 
  Construct the solutions population  $\Omega_D$  from  $\Omega_s$ 
  Repeat
     $t \leftarrow t+1$ 
    If  $\Omega_D = \emptyset$  Then initialize  $P_s$  randomly with static nets
    Else
      Complete  $\Omega_D$  with valid solutions
      Select best solutions
      Compute Entropy function
      Compute adequate probabilities Entropy-based
      If ( $T$  converges to  $T_{\max}$ ) and (all  $p_k$  still negligible)
        Then StateSystem  $\leftarrow$  EquilibriumPoint {Stop evolution}
      Else
        Apply evolutionary operators
        Evaluate score-based  $\Omega_s$ 
        Construct  $\Omega_s$  {  $\Omega_D \equiv \Omega_s$  }
  Until (stateSystem = EquilibriumPoint)

```

4 Experiments

This section presents details of some efficient optimization algorithms used in the goal of comparing different generalization performances reached on data sets.

4.1 Comparison Methods

Estimation of Distribution Algorithms (EDAs). EDAs include a class of algorithms and were first introduced by Mühlenbein and Pâaß [15]. The template of EDAs [13], [16] can be described as follows:

- Initialize population (usually randomly)
- Select promising individuals
- Estimate the distribution of those promising individuals (probabilistic model building)
- According to the distribution (model), sample new solutions, and then go to step 2 until the halting criterion is met.

The following experiments are tested with a simple genetic algorithm (sGA) and three distribution estimation algorithms (DEAs): a compact GA (cGA), an extended compact GA (ecGA), and the Bayesian Optimization Algorithm (BOA). Instead of the mutation and crossover operations of conventional GAs, DEAs use a statistical model of the individuals that survive selection to generate new individuals. Numerous experimental and theoretical results show that DEAs can solve hard problems reliably and efficiently.

4.2 Data Sets

Table 1 introduces briefly the data sets used in the experiments and which are available in the UCI repository [6]. The parameters 1, 2, 5 mentioned with BP (Backpropagation) represent the number of epochs.

4.3 Empirical Results

Table 2 shows that the sGA and the BOA finished in similar number of generations (except for Credit-Australian), and were the slowest algorithms in most cases. On most data sets, the ecGA finishes faster than the other algorithms. However, the ecGA produced networks with inferior accuracy than the other methods or the fully-connected networks in the Housing case. Despite the occasional inferior accuracies, it seems that the ecGA is a good pruning method with a good compromise of accuracy and execution time as reported in [7].

Table 1. Description of the data sets used in the experiments

| | | | Features | | Neural | Networks | |
|-------------------|-------|-------|----------|-------|--------|----------|--------|
| Domain | Cases | class | Cont. | Disc. | Input | Output | Hidden |
| Breast Cancer | 699 | 2 | 9 | - | 9 | 1 | 5 |
| Credit-Australian | 659 | 2 | 6 | 9 | 46 | 1 | 10 |
| Credit-German | 1000 | 2 | 7 | 13 | 62 | 1 | 10 |
| Housing | 106 | 3 | 12 | 1 | 13 | 3 | 2 |
| Ionosphere | 351 | 2 | 34 | - | 34 | 1 | 10 |

We notice that all results provided by ELK are relatively better than those mentioned by others algorithms. This proves that the adaptation learning could improve performance better.

Table 2. Performances classification accuracies obtained for each used algorithm

| Domain | ELK | sGA | cGA | ecGA | BOA | 1 BP | 2 BP | 5 BP |
|-------------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Breast Cancer | 98.85 | 96.54 | 96.13 | 95.84 | 96.42 | 98.48 | 98.91 | 99.03 |
| Credit-Australian | 87.01 | 85.78 | 85.75 | 86.18 | 85.84 | 84.12 | 83.75 | 83.32 |
| Credit-German | 71.00 | 70.68 | 70.92 | 70.30 | 70.14 | 71.15 | 71.98 | 70.12 |
| Housing | 77.36 | 75.36 | 67.11 | 64.18 | 66.24 | 68.77 | 70.91 | 69.54 |
| Ionosphere | 87.88 | 84.61 | 82.95 | 82.22 | 84.22 | 68.43 | 69.43 | 67.86 |

5 Conclusion

In this paper, we introduce a learning framework to training feedforward neural networks. This perspective aided to get a clear vision in neural learning. The used ELK will be parameterize and eventually adapted in order to generalize its application to other neural architectures like Kohonen or radial basis networks. Even more than

an artificial vision the global view of the framework gives a real position in neuroscience. The model proves the rigid relation between the genetic metaphor and the control of brain connectivity. The functionality of each parameter and its granularity in the model ensures to the artificial brain memory the cognitive behavior.

References

1. Abbass, H.A.: An evolutionary artificial neural networks approach for breast cancer diagnosis, *Artificial Intelligence in Medicine*, Vol. 25 (3) (2002) 265-281
2. Abraham, A., Nath, B.: Optimal design of Neural nets using hybrid Algorithms, *Pacific Rim International on Artificial Intelligence* (2000) 510-520.
3. Abraham, A., Nath, B.: An adaptive learning framework for optimizing artificial neural networks, *lectures notes in Computer Science*, Vol. 2074 (2001).
4. Bäck, T., Fogel, D.B., Michalewicz, Z.: *Evolutionary computation 2: advanced algorithms and operators*. Institute of physics publishing (2000)
5. Beyer, H.-G., Deb, K.: On self-adaptive features in real-parameter evolutionary algorithms. *IEEE Transaction on Evolutionary Computation* Vol. 5 (3) (2001) 250-270
6. Blake, C.L, Merz, C.J.: *UCI repository of machine learning databases* (1998)
7. Cantù-Paz, E.: Pruning neural networks with distribution estimation algorithms, in *Genetic and evolutionary conference GECCO 2003*, Springer-Verlag, Berlin (2003) 790-800
8. Conradie, A.E., Miiikkulainen, R., Aldrich, C.: *Intelligent Process Control Utilizing Symbiotic Mimetic in Proceeding. of 2002 Congress on Evolutionary Computation, CEC 02*, Honolulu, HI, USA, vol. 1 (2002) 623-628,
9. Fogel, D. B.: *Evolutionary computing*. IEEE Press (2002)
10. Gepperth, A., Roth, S.: Applications of multi-objective structure optimization. In *13th European Symposium on Artificial Neural Networks (ESANN 2005)* (2005)
11. Hüsken, M.H., Jin, Y.: Sendho B.: Structure optimization of neural networks for aerodynamic optimization. *Soft Computing*, Vol. 9 (1) (2005) 21-28
12. Hiroshi, T., Hisaaki, H.: The functional localization of neural networks using genetic algorithms, *Neural Networks* Vol. 16 (2003) 55–67.
13. Larrañaga, P.: A review on estimation of distribution algorithms. In Larrañaga, P., & Lozano, J. A. (editors), *Estimation of Distribution Algorithms*, Chapter 3, Kluwer Academic Publishers (2002) 57-100.
14. Mohamed Ben Ali, Y., Laskri, M.T.: An evolutionary machine learning: an adaptability perspective at fine granularity, *International Journal of Knowledge-based and Intelligent Engineering Systems* (2005) 13-20.
15. Mühlenbein, H., Pääß, G.: From recombination of genes to the estimation of distributions I. Binary parameters (1996) 178-187
16. Pelikan, M., Goldberg, D. E., Cantù-Paz, E.: BOA: The Bayesian optimization algorithm. *Proceedings of the Genetic and Evolutionary Computation Conference 1999: Volume 1*, San Francisco, CA, Morgan Kaufmann Publishers (1999) 525 –532
17. Rocha, M., Cortez, P., Neves, J.: *Evolutionary Neural Network and Abreu, S., editors, Progress in Artificial Intelligence* (2003)
18. Simi, J., Orponen, P., Antti-Poika, S.: A computational taxonomy and survey of neural network models, *Neural Computation*, Vol. 12 (12) (2000) 2965-2989
19. Stanley, K.O., Miiikkulainen, R.: Evolving neural networks through augmenting topologies, *Evolutionary Computation*, Vol. 10 (2) (2002) 99–127.

20. Stanley, K.O., Miikkulainen, R.: Competitive coevolution through evolutionary complexification, *Journal of Artificial Intelligence research*, Vol. 21 (2004) 63-100.
21. Yao, X.: Evolving artificial neural networks, *proceeding of the IEEE*, Vol. 87 (9) (1999) 1423-1447
22. Wicker, D., Rizki, M., Tamburino, L. A.: E- Net: Evolutionary neural network synthesis, *Neurocomputing*, Vol. 42 (2002) 171-196

Improving the Quality of the Pareto Frontier Approximation Obtained by Semi-elitist Evolutionary Multi-agent System Using Distributed and Decentralized Frontier Crowding Mechanism

Leszek Siwik and Marek Kisiel-Dorohinicki

Department of Computer Science
AGH University of Science and Technology, Kraków, Poland
{siwik,doroh}@agh.edu.pl

Abstract. The paper presents one of additional mechanisms called *distributed frontier crowding* which can be introduced to the *Semi-Elitist Evolutionary Multi Agent System—selEMAS* and which can significantly improve the quality of obtained Pareto frontier approximation. The preliminary experimental comparative studies are based on a typical multi-objective problem presenting the most important features of the proposed approach.

1 Introduction

The approach of *evolutionary multi-agent systems* is both similar as well as different from classical evolutionary algorithms. The key idea of EMAS is the incorporation of evolutionary processes into a multi-agent system at a population level [6]. Thus EMAS may be considered as a computational technique utilizing a *decentralized* model of evolution, unlike (extending) classical evolutionary computation. It was already shown that EMAS-based multiobjective optimization allows for obtaining high quality approximations of Pareto frontiers for both discrete and continuous multi-objective optimization problems [9,7]. It has been also shown that on the basis of the EMAS further research and more sophisticated approaches, models and mechanisms can be proposed [4,8,3]. For instance including the—borrowed from classical evolutionary algorithms—elitism mechanism into an evolutionary multi-agent system in general, and into EMAS for multiobjective optimization in particular, is possible, quite simple in realization, and leads to obtaining quite promising results [8]. It turns out, that model of semi-elitist evolutionary multi-agent system allows for introducing further modifications responsible for improving the quality of the Pareto frontier approximation. The model that is being discussed in this paper introduces a special mechanism, which controls the dispersing of individuals over the Pareto frontier, which is called *distributed frontier crowding* by analogy to well-known niching technique of crowding.

The paper is organized as follows. Section 2 gives the details of the semi-elitist evolutionary multi-agent system—as the basis of proposed here distributed frontier crowding mechanism. In section 3 details of the realizations of the semi-elitist EMAS with distributed frontier crowding are presented. In section 4 the problem used for evaluation of the proposed approach and comparison criteria are presented. Finally, the results obtained

by seEMAS with distributed frontier crowding in comparison to results obtained by EMAS and seEMAS, as well as by MOGA algorithm are discussed in the last section.

2 Semi-elitist EMAS for Multiobjective Optimization

The starting point for considerations about distributed frontier crowding mechanism is a Semi-Elitist Evolutionary Multi-Agent System (seEMAS) [8] for multiobjective optimization that should be perceived as a modification of the Evolutionary Multi-agent System (EMAS) [9]—modification that—according to the presented results—causes significant improvement of the Pareto frontier approximation. Before the distributed frontier crowding mechanism will be described in details in next section—below the seEMAS approach is discussed.

Including elitism into *EMAS* may be obviously done in different ways. In the considered case, it is based on a slightly modified structure of the agents' world (see fig. 1a). In comparison to the structure of the *EMAS* environment (presented for example in [8]) the modification consists in introducing an additional, elitist type of an island, and a special action that can be performed (only) by selected (elitist) agents allowing them for migration to this very island. The exceptionality of the elitist island lies in the dedicated elite-preserving operators that are applied to the elitist agents. Additionally, in the “semi-elitist” island there is no evolution process, and there are only coming in paths to this island(s). Thus agents, which decided to migrate to this island are not able to go back from elitist to “ordinary” island(s), and in consequence they cannot take part in the process of evolution. So, because this model assumes that identified elitist agents do not take part in the evolutionary processes, the approach is called *semi-elitist*, even though such external set storing the best individuals is referred as a kind of elitism in the literature [1] and even though that since the best agents can not be removed from the system—*EMAS* possesses the natural built-in elitist mechanism but an overview of this mechanism is out of the scope of this paper.

Unfortunately, in case of multicriteria optimization *since there are many objective functions, it is not as straightforward as in the single-objective case to identify the elite. In such situations, the non-dominating ranking comes to our rescue* [2, p.240]. In the

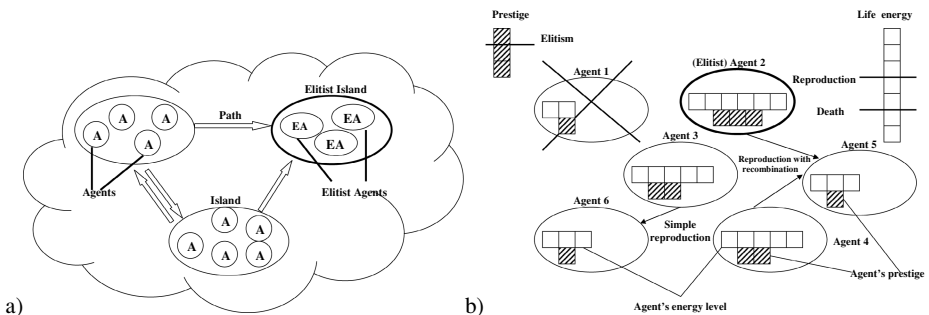


Fig. 1. Structure (a) and resources' thresholds (b) of semi-elitist *EMAS*

described realization the mechanism allowing for identification of the elite is based on the level of additional (in comparison to the "classical" EMAS) resource called *prestige*, which is being gathered (and is not lost) by agents during their life. The level of *prestige* at the beginning of agents' life is equal to *zero*. Then, every time an agent dominates (in the sense of *domination relation*) any other agent, its level of *prestige* increases, and so it may be assumed that agents with high level of this specific resource belong to the elite. This very mechanism allows—in quite elegant, easy to understand and to implement way, that does not require any additional complicated operations and computations—for the realization of the above-mentioned idea of *non-dominating ranking*.

Of course, the proposed mechanism requires the modification (in the comparison to the "classical" EMAS approach of course) of agent's decision-making algorithm, which can be formulated as it is shown in fig. 2a.

| | |
|--|--|
| <pre> Procedure Make_an_elitist_decision { Make_a_decision() if (prestige > elite_threshold) Perform elitist action(s) a) }</pre> | <pre> Procedure Perform_elitist_action() { if (elitist island is not empty) Introduce itself to the elitist agents if (I am dominated by elitist agent(s)) exit if (I dominate elitist agent(s)) Eliminate them from the system Perform simple reproduction Migrate to the elitist island b) }</pre> |
|--|--|

Fig. 2. Make_an_elitist_decision() procedure (a) and Perform_elitist_action() procedure (b)

The only difference between *Make_an_elitist_decision()* procedure and "classical" *Make_a_decision()* procedure (i.e. between elitist and non-elitist agent's decision making) consists in adding an additional decision point and an additional block of operations¹. The additional decision point is connected with checking if an agent belongs to the elite. If so, the agent performs elitist actions.

The elitist action considered in the discussed approach consists in migration to the elitist island. If this island is not empty, an agent tries to introduce itself to all other members of the elite. If the agent discovers during this process that it is dominated by another member of the elite it does not migrate. Next, if the agent meets during this process agents that it dominates, it eliminates them from the system.

After all these meetings, when an agent is sure that it can migrate to the elitist island, it performs a final action in its ordinary island, which is a simple reproduction (i.e. cloning and then applying the mutation operator). Then it migrates to the elitist island and begins (according to the assumption that the elite does not take a part in the process of evolution) the only activity consisting in meetings with agents entering the elitist island. Thus *Perform_elitist_action()* procedure can be formulated as it is shown in fig. 2b.

¹ One may ask why these decisions are made by agent at the end of its decision process. This is because in all species the most important is to survive and to have offspring, and only if these "necessities" are satisfied individual can satisfy its higher needs.

3 Semi-elitist EMAS with Distributed Frontier Crowding

It turns out, that applying the described in the previous section semi-elitist operator(s) into an evolutionary multi-agent system—despite of other advantages—gives the opportunity for introducing additional mechanism(s) responsible for improving dispersing individuals over the Pareto frontier. This can be very useful, especially during solving quite difficult problems with a disconnected Pareto frontier (and Pareto set as well). One of such mechanisms is the technique that might be called *distributed frontier crowding*, because it applies to criteria space—by analogy to the commonly used notion of *crowding* which applies to the solution space (search domain). Such mechanism can be realized as follows: During the meetings, an agent is able to check if a solution represented by its opponent is located in its surrounding (i.e. if the distance between solutions represented by the agents is less than the given value r). If so, an agent increases its personal counter storing the number of individuals located in the same fragment of the objective space. Additionally, asking its opponents for the number of individuals located in their surroundings, an agent is able to gather with time the (partial) knowledge about the average number of individuals located in areas represented by met agents. Thanks to this, if an agent becomes the elitist one, it can make a decision about migrating to the elitist island on the basis of the relation between the number of individuals located in its surrounding and the average number of individuals located in other regions of the objective space.

```

Procedure Perform_distributed_frontier_crowding_elitist_action() {
  if (elitist island is not empty)
    Introduce itself to the elitist agents
    if (I am dominated by elitist agent(s)) exit
    if (number of neighbors > avg. number of agents in other areas of the frontier)
      if (number of my neighbors < m * avg. number of agents in other areas of the PF)
        Perform simple reproduction action with macromutation
      Migrate to the elitist island
      if (I dominate elitist agent(s))
        Eliminate them from the system
    else
      if (k * number of my neighbors < avg. number of agents in other areas of the PF)
        Perform simple reproduction with (micro)mutation
  }

```

Fig. 3. Perform_distributed_frontier_crowding_elitist_action() procedure

Of course such modified decisive process can be realized in many different ways. In the simplest form, an agent may decide about migration if the number of its direct neighbors is greater than the average number of individuals located in another areas. Moreover, if an agent discovers that number of its direct neighbors is much lesser than the number of individuals located in other areas of objective space, it can make a decision about reproduction with mutation (but this time mutation operator is applied with

a very small range—sampling promising areas of the objective space). So, a modified *Perform_elitist_action()* that takes into consideration the described above mechanism can be formulated as it is shown in fig. 3.

Such mechanism (assuming $k > m$ or even $k \gg m$) allows for: deeper exploration of the areas of the Pareto frontier which are worse sampled than its other regions, maintaining the diversity of agents population and thus—avoiding evolution stagnation.

4 Testing Problems and Comparison Criteria

The experimental and comparative studies are based on known from the literature testing problem—i.e. on the so-called *Kursawe* problem. Its definition is as follows:

$$Kursawe = \begin{cases} f_1(x) = \sum_{i=0}^{n-1} (-10 \exp(-0.2 \sqrt{x_i^2 + x_{i+1}^2})) \\ f_2(x) = \sum_{i=1}^n |x_i|^{0.8} + 5 \sin x_i^3 \\ n = 3 \quad -5 \leq x_1, x_2, x_3 \leq 5 \end{cases}$$

and in this case optimization algorithm has to deal with disconnected two-dimensional Pareto frontier and disconnected three dimensional Pareto set. Additionally, a specific definition of f_1 and f_2 functions causes that even very small changes in the space of decision variables can cause big differences in the space of objectives. All of these cause that *Kursawe* problem is quite difficult for solving in general—and for solving using evolutionary techniques in particular.

In the literature there at least three criteria distinguishing high-quality approximation of Pareto frontiers can be found: the closeness to the model Pareto frontier (the closer the better), the number of individuals belonging to the proposed approximation (the more the better), and the dispersing of individuals over the whole Pareto frontier [2]. All these criteria will be considered in the comparison below.

To present some reference points allowing for evaluation of the results obtained by the proposed agent-based technique, two algorithms will be used i.e. "classical" (non-elitist) evolutionary multi-agent system and a classical (i.e. non agent-based) evolutionary algorithm for multiobjective optimization. Thus some figures presented in the section 5 include also results and characteristics obtained using EMAS and MOGA—*Multiple Objective Genetic Algorithm* [5] as well.

5 EMAS vs selEMAS with *Distributed Frontier Crowding*—Comparative Studies

As it was mentioned in section 4 the influence of proposed in this paper mechanism was tested using, inter alia, three dimensional problem with disconnected Pareto frontier and Pareto set as well—i.e. so called *Kursawe* problem. But proposed approach was

² A similar mechanism can be based on the space of decision variables, which should cause (and causes in fact) deeper exploration of these areas of the Pareto set, which are worse sampled than its other regions—preliminary results are quite promising but it requires further research that goes beyond the scope of this article.

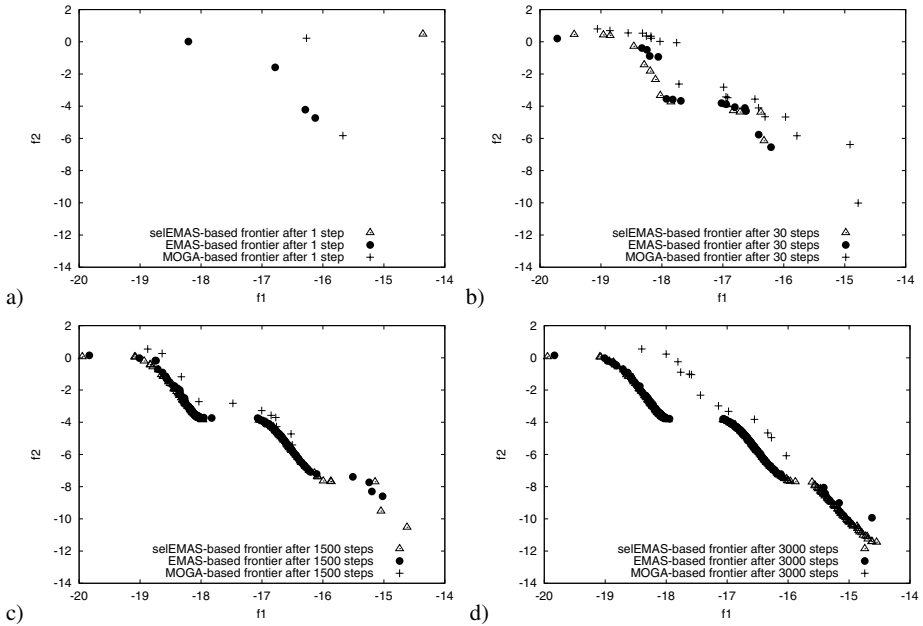


Fig. 4. Pareto frontier approximations for *Kursawe* problem obtained by EMAS-, selEMAS and MOGA-based algorithms after 1 (a), 30 (b), 1500 (c) and 3000 (d) steps

tested also using both not so difficult problems (*MaxEx*), and more difficult problems (*ZDT-4*, *ZDT-5*, *ZDT6*). In case of simple problems with coherent Pareto sets and Pareto frontier as well (such as *MaxEx* problem [2]) the crucial influence of proposed here frontier crowding mechanism on the Pareto frontier approximation was not observed. Generally it can be said that proposed mechanism seems to be the more important the more difficult is the problem that should be solved.

In fig. 4 there are presented approximations of the Pareto frontier obtained by analyzed algorithm (and—for comparison—by MOGA and EMAS as well) after 1, 30, 1500 and 3000 steps. As one may see MOGA is definitely the worst alternative—since it does not allow for obtaining satisfactory solution. Unfortunately in this case (i.e. in case of *Kursawe* problem) also both agent-based algorithms are not able to find high-quality approximation of the Pareto frontier. Generally, analysis of fig. 4a,b and c, and fig. 6 as well, confirms that EMAS and selEMAS behave similarly and allow for obtaining very similar approximations of the Pareto frontier. These approximations are better, as the matter of fact, than the one obtained by MOGA algorithm—but they still leave a lot to be desired. Fortunately, it seems that presented in section 3 *distributed frontier crowding* mechanism comes to our rescue.

To present the influence of this mechanism on solutions that are being obtained, the following experiment has been performed. Solving *Kursawe* problem for the first 1500 steps of system run the *distributed frontier crowding* was switched off, then until the end of computations this mechanism was switched on. In the consequence—as it was mentioned—initially (i.e. during first ca. 1500 steps of system run) both algorithms

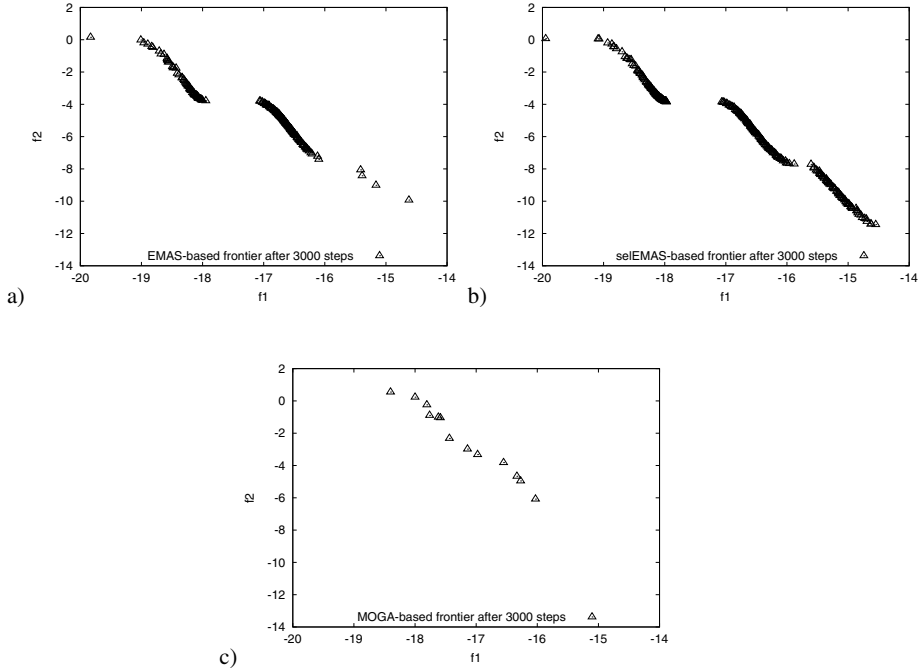


Fig. 5. Pareto frontier approximations for *Kursawe* problem obtained by EMAS (a), seEMAS (seEMAS with *distributed frontier crowding* between 1500th and 3000th step of system run) (b) and MOGA (c) algorithms after 3000 steps of system simulation

(i.e. *Evolutionary Multi-Agent System* and *Semi-Elitist Evolutionary Multi-Agent System* without *frontier crowding*) behave very similarly (see fig. 6) and are quite similarly effective (see fig. 4a–c).

Switching on the *distributed frontier crowding* has obvious influence on almost all characteristics, what explains such violent and significant changes of values presented in fig. 6. But what is the most important, after comparing fig. 4c and fig. 4d there is no doubt that switching on the *distributed frontier crowding* has significant influence on the quality of obtained by system Pareto frontier since—in contrast to the EMAS-based system and seEMAS-based system without *frontier crowding*—this very algorithm is able to discover all parts of the Pareto frontier and additionally within all these parts no-dominated individuals are evenly dispersed.

Because in fig. 4 (especially in fig. 4d) presented Pareto frontiers (partially) overlap, in fig. 5 there are separately presented approximations of the Pareto frontier and Pareto set obtained by seEMAS, EMAS and MOGA algorithm after 3000 steps (in the case of seEMAS it was the system with *distributed frontier crowding*). After analyzing these figures there is no doubt that seEMAS algorithm with *distributed frontier crowding* is definitely the best alternative.

In fig. 6 there are presented some characteristics describing behavior of analyzed system. In fig. 6a there are presented size of EMAS and seEMAS population in

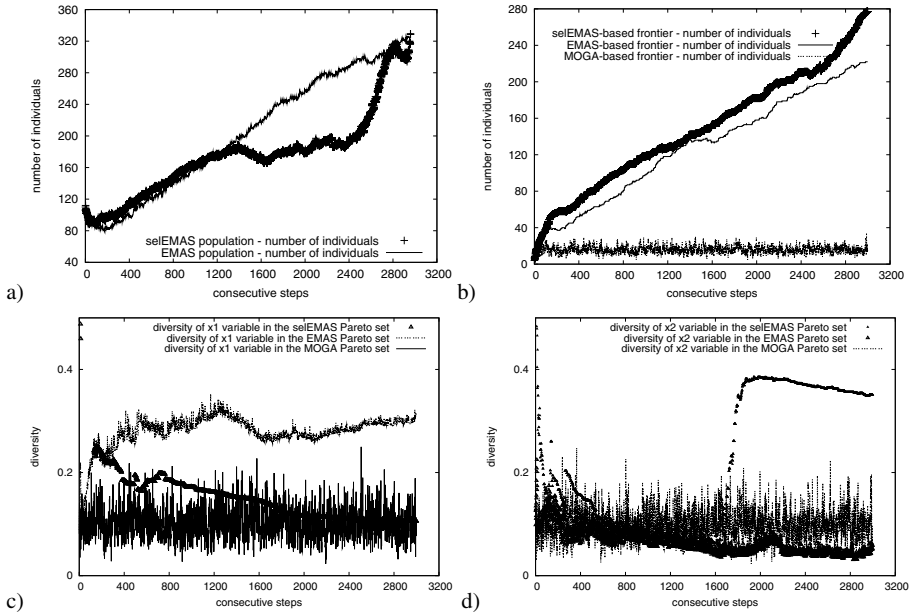


Fig. 6. Selected characteristics obtained during solving *Kursawe* problem: size of EMAS and seLEMAS (seLEMAS with *distributed frontier crowding* between 1500th and 3000th step of system run) population (a), number of found non-dominated solutions (b), and diversity of x_1 (c), and x_2 (d) decision variable in EMAS- seLEMAS- and MOGA-based Pareto set

consecutive steps of system run. As one may see—during first 1500 steps both populations have almost the same size—then the size of seLEMAS population significantly decreases—which is caused by switching on the *distributed frontier crowding*. It is worth to pay attention to the fig. 6b presenting number of non-dominated solutions found by both agent-based algorithms and by MOGA as well. As one may notice—during the whole simulation—non-dominated seLEMAS-based frontier is more numerous than the EMAS-based one. And with time, especially during last stage of simulation this difference is deeper and deeper (to seLEMAS advantage of course). In fig. 6b there is also presented the size of the Pareto frontier obtained by MOGA—and as it is shown MOGA only during first a dozen (or a few dozen) or so steps is as effective as agent-based methods. Additionally, analyzing only sizes of populations and Pareto frontiers, it can be said that during solving *Kursawe* problem seLEMAS algorithms (especially with *distributed frontier crowding*) turned out to be more efficient since it is able to discover more non-dominated solutions using significantly less numerous population (such characteristic took place also during another experiments).

Based on the presented results and characteristics it can be said that taking efficiency into consideration a kind of hybrid system based on seLEMAS can be proposed. Assuming N steps of such system run—during first $N/2$ the *distributed frontier crowding* can be switched off—what is conducive to drifting non-dominated solutions to the model

Pareto frontier. Then, switching on the *frontier crowding*—causes as the matter of fact more complexity of the system—but simultaneously it allows for much better dispersing individuals over the Pareto frontier.

6 Concluding Remarks

Presented results confirm that semi-elitist evolutionary multi-agent system approach allows for introducing additional mechanism(s) responsible for "controlling" and significant improving the dispersing of individuals over the Pareto frontier. Presented in this paper *distributed frontier crowding* mechanism is only one of such mechanism(s), but as it was shown, introducing this technique significantly improves the quality of the Pareto frontier approximation (see fig. 4c and fig. 4d in section 5). The natural extension of *distributed frontier crowding* is applying similar mechanism in the space of decision variables. However, although preliminary results prove in the promising way, that such mechanism allows for deeper exploration of the Pareto set—it needs further research.

To recapitulate this article, presented results, our approach and motivation—there are several 'because's' that have to be emphasized in this place, i.e.:

- Because this paper is devoted to improving EMAS-based multiobjective optimization (not to introducing a new general mechanisms responsible for dispersing solutions over the Pareto frontier) in the course of the paper obtained results are compared not to results obtained using another "disperse-preserving" mechanisms (such as niching, incremental learning, crowding distance etc.) but rather to results obtained by Evolutionary Multi-Agents Systems that have been deprived any additional mechanisms of this kind (in another words this paper—and our research at the moment is devoted to improving EMAS-based multiobjective optimization not to introducing a new crowding mechanism).
- Because the EMAS-based approach to multiobjective optimization is not very popular—authors decided to present it in details. In the consequence, because of the space limitations only qualitative results are here presented. The quantitative results i.e. the value of some metrics—such as HV, HVR, spread etc.—will be discussed in details in the next article related to conducted research.
- Because of the readability of the presented results—statistical characteristics (such as standard deviation etc.) have been omitted—however presented results have been recurrent during conducted experiments and representative characteristics are here presented.

In the next paper related to this topic apart from the quantitative results also the formal mathematical model of the proposed approach will be presented. Simultaneously further experiments taking into considerations multiobjective problems with more dispersed Pareto frontier and combinatorial multiobjective problems will be conducted.

It was mentioned as the matter of fact that—at the moment at least—authors are focusing on Evolutionary Multi-Agent System not on introducing new disperse-preserving mechanisms—but especially because such characteristics as: almost complete transparency related to computational complexity—the further research devoted

to proposed distributed frontier crowding mechanism will be also conducted. For example presented mechanism needs further research related to the value of m and k parameters (see the `Make_distributed_frontier_crowding_elitist_action()` procedure in section 3) etc.—especially that some problems with stabilization of the population size have been observed.

References

1. C. Coello Coello and G. Toscano. Multiobjective structural optimization using a micro-genetic algorithm. *Structural and Multidisciplinary Optimization*, 30(5):388–403, November 2005.
2. K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, 2001.
3. R. Drezewski and L. Siwik. Co-evolutionary multi-agent system with sexual selection mechanism for multi-objective optimization. In *Proceedings of the IEEE Congress on Evolutionary Computations, CEC2006*, number 0-7803-9487-9, pages 769–776. IEEE, July 2006.
4. R. Drezewski and L. Siwik. Multi-objective optimization using co-evolutionary multi-agent system with host-parasite mechanism. In V. Alexandrov, P. van Albada, G.D. amd Sloot, and J. Dongarra, editors, *Proceedings of the 6th International Conference Computational Science - ICCS 2006, PartIII*, volume 3993 of *LNCIS*, pages 871–878. Springer, May 2006.
5. C. Fonseca and P. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Genetic Algorithms: Proceedings of the Fifth International Conference*, pages 416–423. Morgan Kaufmann, 1993.
6. M. Kisiel-Dorohinicki. Agent-oriented model of simulated evolution. In W. I. Grosky and F. Plasil, editors, *SofSem 2002: Theory and Practice of Informatics*, volume 2540 of *Lecture Notes in Computer Science*. Springer-Verlag, 2002.
7. L. Siwik and M. Kisiel-Dorohinicki. Balancing of production lines: evolutionary, agent-based approach. In *Proceedings of Conference on Management and Control of Production and Logistics*, pages 319–324, 2004.
8. L. Siwik and M. Kisiel-Dorohinicki. Semi-elitist evolutionary multi-agent system for multi-objective optimization. In V. Alexandrov, P. van Albada, G.D. amd Sloot, and J. Dongarra, editors, *Proceedings of the 6th International Conference Computational Science - ICCS 2006, PartIII*, volume 3993 of *LNCIS*, pages 831–838. Springer, May 2006.
9. K. Socha and M. Kisiel-Dorohinicki. Agent-based evolutionary multiobjective optimisation. In *Proc. of the 2002 Congress on Evolutionary Computation*. IEEE, 2002.

On Semantic Properties of Interestingness Measures for Extracting Rules from Data

Mondher Maddouri¹ and Jamil Gammoudi²

¹ Department of Maths & Computer Sciences,
National Institute of Applied Sciences & Technology of Tunis – INSAT
University of Carthago,
Centre Urbain Nord, B.P. 676, 1080 Tunis Cadex – Tunisia
mondher.maddouri@fst.rnu.tn

² Department of Computer Sciences,
Faculty of Law, Economics and Management of Jendouba – FSJEG
University of Jendouba,
Avenue de l'UMA - 8189 Jendouba – Tunisia
gammoudij@gmail.com

Abstract. The extraction of IF-THEN rules from data is a promising task of data mining including both Artificial Intelligence and Statistics. One of the difficulties encountered is how to evaluate the relevance of the extracted rules? Many authors use statistical interestingness measures to evaluate the relevance of each rule (taken alone). Recently, few research works have done a synthesis study of the existing interestingness measures but their study presents some limits. In this paper, firstly, we present an overview of related works studying more than forty interestingness measures. Secondly, we establish a list of nineteen other interestingness measures not referenced by the related works. Then, we identify twelve semantic properties characterizing the behavior of interestingness measures. Finally, we did a theoretical study of sixty two interestingness measures by outlining their semantic properties. The results of this study are useful to the users of a data-mining system in order to help them to choose an appropriate measure.

1 Introduction

In Data Mining, Data knowledge extracted is generally represented by a whole of rules (classification or association rules). The first method allowing the generation of the association rules was proposed in 1993 by Agrawal [1] through its APRIORI algorithm. Indeed, the use of APRIORI algorithm in Data Mining makes it possible to test the various possible combinations between the exogenous variables (Attributes) to find potential relationships (associations) which will be expressed in the form of association rules. However, the rules selected by APRIORI algorithm are evaluated by the support and confidence. These two interestingness measures allow evaluation of association rules quality and to evaluate their relevance. The advantage of this algorithm is that it is unsupervised (not preliminary knowledge about classes). However, it has a disadvantage by the exponential number of generated rules [6].

In order to solve this limit, several approaches were proposed. In fact, we can classify these approaches in two classes. The first one is purely algorithmic which is the improvement or the extensions of APRIORI algorithm, through heuristics in the objective to reduce the space of research and/or to perform the Data Mining process [15]. The second class relates to the post-processing of the rules generated by APRIORI algorithm. These suggested approaches help users to choose the best rules which are adapted to their data and their needs. According to Huynh [12] some research works related to the approaches of post-processing are summarized as following:

- Advanced techniques of visualization in order to improve the legibility of the rules [6],
- Interactive techniques for searching summaries [11],
- Techniques to reduce the redundancies and preserving non-redundant rules (logical implication) [15].
- Optimization techniques by a Multi-criteria Decision Aid [16]
- Complementary interestingness measure to evaluate association rules [7].

In our work, we are interested to the last mentioned approach. In fact, we propose, firstly, to draw up an "exhaustive" list of the various measures suggested in the literature. Secondly, we identify semantic properties of interestingness measures in the objective to characterize these measures. This task will be exploited by the user to choose the appropriate measure which respond at his needs in function of the studied data.

In this paper, we present an overview of related works to our contribution in the first section. In the second section, we present interestingness measures that are not considered. The third handles the added properties which are dispersed in several works. At the last section, we study the behaviour of each measure in reference to its semantic properties in the objective to compare them.

2 The Interestingness Measures: An Overview

In the two last decades, many statistical measures were introduced to select relevant rules (Hypothesis) in learning processes [6], [12]. Several application domains have contributed to the diversification of interestingness measures of association rules. Each suggested new measure responds to one or more needs in the form of constraints which are dependent on the data or on the preferences specified by the users.

Let $A \rightarrow B$ be an association rule. The different interestingness measures are defined using the parameters $p(AB)$, $p(\overline{AB})$, $p(A\overline{B})$, $p(\overline{A}B)$ where $p(X)$ represents the probability of X .

In fact, in related works, some attempts of synthesis were carried out. They consist, in a first stage, to define semantic properties allowing characterizing the measures. In a second stage, the authors try to compare these measures by classifying them according to their experimental behaviour.

To make a comparison of these measures and to study their behavior, some authors were interested to propose properties judged interesting to characterize an interestingness measure [17], [16]. In a first work, three properties were proposed in [17]. After that, eight properties were suggested in [16]. Other properties were proposed in similar works, but in specific contexts (biologic data and textual data). In [2], the author is interested in unexpected and stable rules obtained from noisy data. Consequently, they focus on insensitive measures to the noises that can guarantee the selection of the most stable rules.

In the following section we present the most interesting work which is related to our contribution.

2.1 Properties Study for Multi-criteria Decision Support

In [16], Lallich and his team propose a different way to resolve the problem of association rule selection. Indeed, they present a multi-criteria decision support system to satisfy the needs and the preferences of the user.

Table 1. Interestingness measures studied in [16]

| | Measure | Expression |
|-----|--|---|
| m1 | <i>Support</i> | $p(AB)$ |
| m2 | <i>Confidence</i> (precision/accuracy) | $p(B/A)$ |
| m3 | <i>Pearson's correlation coefficient</i> | $\frac{p(AB) - p(A)p(B)}{\sqrt{p(A)p(B)p(\bar{A})p(\bar{B})}}$ |
| m4 | <i>Centered confidence</i> | $p(B/A) - p(B)$ |
| m5 | <i>PS (Novelty, RI)</i> | $n \cdot p(A)(p(B/A) - p(B))$ |
| m6 | <i>Loevinger (Certainly factor)</i> | $\frac{p(B/A) - p(B)}{p(\bar{B})}$ |
| m7 | <i>Zhang</i> | $\frac{p(AB) - p(A)p(B)}{\max\{p(AB)p(\bar{B}), p(B)p(\bar{A}) - p(AB)\}}$ |
| m8 | <i>-(implication index)</i> | $-\sqrt{n} \frac{p(\bar{A}\bar{B}) - p(A)p(\bar{B})}{\sqrt{p(A)p(\bar{B})}}$ |
| m9 | <i>Lift (Interest Factor))</i> | $\frac{p(A \cap B)}{p(A)p(B)}$ |
| m10 | <i>Least contradiction</i> | $\frac{p(AB) - p(\bar{A}\bar{B})}{p(B)}$ |
| m11 | <i>Sebag-Schoenauer index</i> | $\frac{p(AB)}{p(\bar{A}\bar{B})}$ |
| m12 | <i>Odd multiplier</i> | $\frac{p(AB)p(\bar{B})}{p(\bar{A}\bar{B})p(B)}$ |
| m13 | <i>Conviction</i> | $\frac{p(A)p(\bar{B})}{p(\bar{A}\bar{B})}$ |
| m14 | <i>Example and counter-example ratio</i> | $\frac{p(AB) - p(\bar{A}\bar{B})}{p(AB)}$ |
| m15 | <i>Kappa Coefficient (Cohen's quality index)</i> | $\frac{p(AB) + p(\bar{A}\bar{B}) - p(A)p(B) - p(\bar{A})p(\bar{B})}{1 - p(A)p(B) - p(\bar{A})p(\bar{B})}$ |
| m16 | <i>Information gain</i> | $\log_2 \left(\frac{p(A \cap B)}{p(A)p(B)} \right)$ |
| m17 | <i>intensity of Implication</i> | $P \left[\text{Poisson} \left(n \cdot p(A)p(\bar{B}) \right) \geq n \cdot p(A)p(\bar{B}) \right]$ |
| m18 | <i>Entropic intensity of implication</i> | $\left\{ \left[\left(1 - h_1(p(AB))^2 \right) \left(1 - h_2(p(AB))^2 \right) \right]^{1/4} \right\}^{1/2}$ |
| m19 | <i>Probabilistic discriminate index (PDI)</i> | $p \left[N(0,1) > \text{IMPIND}^{RC/B} \right]$ |
| m20 | <i>Laplace</i> | $\frac{n \cdot p(A \cap B) + 1}{n \cdot p(A) + 2}$ |

Table 2. Interestingness measures studied in [9]

| | Measure | Expression |
|-----|---|---|
| m21 | <i>Ralambrodrainy</i> | $p(\overline{AB})$ |
| m22 | <i>Descriptive-Confirm</i> | $p(AB) - p(\overline{AB})$ |
| m23 | <i>Wang index</i> | $p(A)p(B/A) - \alpha$ |
| m24 | <i>Descriptive Confirmed-Confidence (Ganascia Index)</i> | $p(B/A) - p(\overline{B}/A)$ |
| m25 | <i>Causal support</i> | $p(AB) - p(\overline{AB})$ |
| m26 | <i>Causal-Confidence</i> | $\frac{1}{2} [p(B/A) + p(\overline{A}/\overline{B})]$ |
| m27 | <i>Causal-Confirmed confidence</i> | $\frac{1}{2} [p(B/A) + p(\overline{A}/\overline{B})] - p(\overline{B}/A)$ |
| m28 | <i>Pearl</i> | $ p(AB) - p(A)p(B) $ |
| m29 | <i>Ratio of link</i> | $\frac{p(AB)}{p(A)p(B)} - 1$ |
| m30 | <i>Cosine(Ochia index)</i> | $\frac{p(AB)}{\sqrt{p(A)p(B)}}$ |
| m31 | <i>Odd's ratio</i> | $\frac{p(AB)p(\overline{AB})}{p(\overline{AB})p(\overline{AB})}$ |
| m32 | <i>Yule's Q (indice de Yule)</i> | $\frac{p(AB)p(\overline{AB}) - p(\overline{AB})p(\overline{AB})}{p(AB)p(\overline{AB}) + p(\overline{AB})p(\overline{AB})}$ |
| m33 | <i>Yule's Y</i> | $\frac{\sqrt{p(AB)p(\overline{AB})} - \sqrt{p(\overline{AB})p(\overline{AB})}}{\sqrt{p(AB)p(\overline{AB})} + \sqrt{p(\overline{AB})p(\overline{AB})}}$ |
| m34 | <i>Jaccard</i> | $\frac{p(AB)}{p(A) + p(B) - p(AB)}$ |
| m35 | <i>Klogsen</i> | $\sqrt{p(AB) \cdot (p(B/A) - p(B))}$ |
| m36 | <i>Interestingness</i> | $\left[\left(\frac{p(AB)}{p(A)p(B)} \right)^k - 1 \right] \cdot (p(A)p(B))^m$ |
| m37 | <i>Mutual Information Ratio</i> | $\sum_{U \in \{A, \overline{A}\}} \sum_{V \in \{B, \overline{B}\}} p(U V) \log_2 \left(\frac{p(U V)}{p(U)p(V)} \right)$ |
| m38 | <i>J-measure</i> | $p(AB) \log \frac{p(AB)}{p(A)p(B)} + p(\overline{AB}) \log \frac{p(\overline{AB})}{p(\overline{A})p(\overline{B})}$ |
| m39 | <i>Gin index</i> | $p(A) \left(p(B A)^2 + p(\overline{B}/A)^2 \right) + p(\overline{A}) \left(p(B/\overline{A})^2 + p(\overline{B}/\overline{A})^2 \right) - pB^2 - p(\overline{B})^2$ |
| m40 | <i>Lerman similarity index</i> | $\sqrt{n} \frac{p(AB) - p(A)p(B)}{\sqrt{p(A)p(B)}}$ |
| m41 | <i>Probabilistic error of Chi2</i> | $1 - p(\chi^2 \leq \chi^2) \geq 1 - \epsilon$ |
| m42 | <i>Information Ratio Contraposé (TIC)</i> | $IRC(A, B) = \sqrt{IR(A \rightarrow B) \cdot IR(\overline{B} \rightarrow \overline{A})}$ |
| m43 | <i>Surprisingness</i> | $\sum_{i=1}^p IG(\epsilon; A_i)$ |

In this work, the authors studied twenty interestingness measures (see table 1) taking into account eight properties. These properties are supposed to be relevant since they include most of the preferences expressed by users [16]. These properties are: asymmetric processing of rules, sensibility to the size of the consequent, reference

situation of independence between antecedent and consequent, reference situation of logical implication, linearity, sensitivity to the data size, easiness to fix a threshold and intelligibility.

This work, has studied measures based on eight properties which are not studied everywhere. Furthermore, in this work they take into account the preference of the user for measure-choice in the context of multi-criteria decision support. However, they studied only twenty measures and there are a lot of other measures which are not considered such as: J-measure [9], etc.

2.2 Measures Study by Guillet

In [9], a synthesis of approximately 43 measures including those studied in [16] was made (see table 2). In this synthesis Guillet studies some properties of the various interestingness measures. Indeed, these properties correspond to: semantic of measure, number of parameters to which the measure is sensitive, linearity of measure, value in independence, value in balance, symmetry of measure, nature of measure (statistical or descriptive measure) and maximum value (limited or not).

In this work, a whole synthetic study is done on interestingness measures. In fact, for each measure they establish some semantic properties. Furthermore, they developed the ARQAT platform which helps the user to evaluate and compare visually the behaviour of thirty six measures.

However, there are other interestingness measures which are not considered, such as: recall [2], gain measure [10], etc. Even for the considered measures Guillet does not establish all the considered properties as the symmetry of the Descriptive-Confirm measure [13].

3 The New Studied Measures

In this section, we try to establish a more exhaustive theoretical study of interestingness measures. For this reason, we report more than sixty measures and we try to establish their behaviour according to semantic properties. We consider the 43 measures reported in [9] which take into account the 20 measures reported in [16]. Furthermore, we have identified 19 new measures (table 3) which are presented and studied partially in [2], [6] and [3].

4 The Studied Semantic Properties

According to [16], the considered semantic properties are insufficient. The authors encounter a difficulty in evaluating these properties. The considered properties are very different according to their kinds and values (numerical or symbolic, ordinal or not, etc). Another difficulty is to express user criteria based on the considered semantic properties. They found difficulty to evaluate the subjective properties such as the simplicity of implementation.

Table 3. New studied Interestingness measures

| | Measure | Expression | Reference |
|-----|--|---|-----------|
| m44 | <i>Recall(sensitivity)</i> | $p(A/B)$ | [2], [6] |
| m45 | <i>dependency</i> | $ (p(B/A) - p(B)) $ | [6], [13] |
| m46 | <i>Satisfaction</i> | $\frac{p(\bar{B}) - p(\bar{B}/A)}{p(\bar{B})}$ | [14] |
| m47 | <i>Specificity</i> | $p(\bar{A}/\bar{B})$ | [14] |
| m48 | <i>Gain</i> | $p(AB) - \theta \cdot p(A)$ | [10] |
| m49 | <i>Ion</i> | $Ion(A \rightarrow B) = \frac{p(B/A) - p(B)}{1 - p(B)} \text{ if A favourite B}$ $Ion(A \rightarrow B) = \frac{p(B/A) - p(B)}{p(B)} \text{ if B favourite A}$ | [19] |
| m50 | <i>Rectangular Gain</i> | $p(AB) \cdot \ A \vee B\ - (p(AB) + \ A \vee B\)$ | [7] |
| m51 | <i>Negative reliability</i> | $p(\bar{B}/\bar{A})$ | [14] |
| m52 | <i>Rogers and Tanimoto index</i> | $\frac{1 - p(\bar{A}\bar{B}) - p(\bar{A}\bar{B})}{1 + p(AB) + p(\bar{A}\bar{B})}$ | [3] |
| m53 | <i>Dice index</i> | $\frac{p(AB)}{p(AB) + \frac{1}{2} (p(\bar{A}\bar{B}) + p(\bar{A}\bar{B}))}$ | [3] |
| m54 | <i>Kulczynski index</i> | $\frac{1}{2} \left(\frac{p(AB)}{p(A)} + \frac{p(AB)}{p(B)} \right)$ | [3] |
| m55 | <i>Collective strength</i> | $\frac{p(AB) - p(\bar{A}\bar{B})}{p(A)p(B) + p(\bar{A})p(\bar{B})} \times \frac{1 - p(A)p(B) + p(\bar{A})p(\bar{B})}{1 - p(AB) - p(\bar{A}\bar{B})}$ | [3] |
| m56 | <i>Shannon conditional entropy</i> | $- p(B/A) \log_2(p(B/A)) - p(\bar{B}/A) \log_2(p(\bar{B}/A))$ | [3] |
| m57 | <i>Quadratic entropy</i> | $p(B/A) [1 - p(B/A)] + p(\bar{B}/A) [1 - p(\bar{B}/A)]$ | [18] |
| m58 | <i>Information ratio(DIR)</i> | $1 - \frac{\hat{E} \left(\frac{p(A \cap \bar{B})}{p(A)} \right)}{\hat{E} \left(p(\bar{B}) \right)}$ | [3] |
| m59 | <i>Probabilistic measure of deviation from equilibrium(IPEE)</i> | $p \left(N(0,1) > \frac{\tilde{n}}{ab} \right)$ | [3] |
| m60 | <i>Quotient</i> | $1 - \frac{\sup \{AB\}}{\text{conf} \{AB\}} \text{ , if } \sup \{AB\} > \text{conf} \{AB\}$ $1 - \frac{\text{conf} \{AB\}}{\sup \{AB\}} \text{ , Otherwise}$ | [5] |
| m61 | <i>Improvement</i> | $\left 1 - \frac{\text{conf} \{AB\}}{\sup \{AB\}} \right $ | [5] |
| m62 | <i>Putative causal dependency</i> | $\frac{1}{2} p(B/A) - p(B) + p(\bar{B}/\bar{A}) - p(\bar{A}) - p(\bar{B}/A) - p(\bar{B}) - p(A/\bar{B}) - p(A) $ | [13] |

In our work, we try to establish a more exhaustive list of semantic properties. For this reason, we report the 8 theoretical and semantic properties that were considered in [16]. This includes the asymmetric processing, the sensibility to the number of positive examples, the reference situation of independence, the reference situation of

logical implication, the linearity (sensibility to the number of counter-examples), the sensibility to the data size, the easiness to fix a threshold and the intelligibility.

Then, we report two properties which were introduced in [2] and [17]. We consider the sensibility to A and the sensibility to the noise. Finally, we propose to consider two new semantic properties indicating the sensitivity to the two reference situations: The incompatibility and the equilibrium.

In the following, we define the twelve semantic properties that we propose to study.

p1:Asymmetric processing of A and B: This property expresses the fact that the rules $A \rightarrow B$ and $B \rightarrow A$ are different [16]. We should not use measures which evaluate these two rules in the same way. Consider that m is a quality measure, we should have:

$$m(A \rightarrow B) \neq m(B \rightarrow A)$$

We will use the value 1 to indicate the asymmetry behaviour and the value 0 for the symmetric behaviour.

p2:Sensibility to B: the interest of a rule depends on the size of B [17]. The measure should be a decreasing function according to the size of B (or $p(B)$). We will use the value 1 when the measure decreases according $p(B)$ and the value 0 for other behaviours.

p3:Sensibility to A: the interest of a rule depends on the size of A [2]. In the same way for B , the measure should decrease when $p(A)$ increases. We will use the value 1 when the measure decreases according $p(A)$ and the value 0 for other behaviours.

p4:Sensitivity to the reference situation of independence: The situation of independence consists on that the realization of B is independent of the realization of A :

$$p(AB) = p(A) \cdot p(B)$$

There is no new information from a rule $A \rightarrow B$, if A and B are independent [17]. It is more important that the value of the measure be constant if A and B are independent. Thus, the measure can help us to identify this reference situation. We will use the value 1 when the measure has a constant value and the value 0 other ways.

p5:Sensitivity to the reference situation of logical implication: When the number of counter examples is null, there is a logical implication indicated by:

$$p(AB) = p(A)$$

In this situation it is desirable that the measure's value be constant. We will use the value 1 when the measure has a constant value and the value 0 other ways.

p6:Sensitivity to the reference situation of equilibrium: When the number of examples is equal to the number of counter examples, there is a situation of equilibrium indicated by:

$$p(AB) = p(\overline{AB})$$

In this case, it is desirable that the measure has a constant value. We will use the value 1 when the measure has a constant value and the value 0 other ways.

p7:Sensitivity to the reference situation of incompatibility: When the number of examples containing both A and B is null, there is a situation of incompatibility indicated by:

$$p(AB) = 0$$

In this case, it is desirable that the measure has a constant value. We will use the value 1 when the measure has a constant value and the value 0 other ways.

p8:Linearity with the number of counter examples around 0^+ : A nonlinear measure is insensitive to the noise and to the appearance of counter examples [16]. In fact, the authors suggest that the reduction of measure's values should be insignificant at the beginning and more important after that (a convex shape) [8]. This weak decrease can be tolerated by the user and that can be explained by the noise or errors of observations. We will use the value 1 when the measure has a linear shape around 0^+ , the value 0 when it has a concave shape and the value 2 when it has a convex shape.

p9:Sensibility to the data size: the rule's interest increases according to data size (n). So, it is desirable that the measure is increasing according to n . But it is possible that the measure provides many rule-values close to the measure maximal-value. So, the measure loses its discriminate capability. We will use the value 1 to indicate the increasing behaviour and the value 0 other ways.

p10:Facility to fix a threshold: this property is important to easily be able to fix the threshold from which we can judge the relevance or not of a rule. This depends on the distribution of the rule-values within the interval of values taken by each measure.

p11:Sensitivity to the noise: It is rare to use perfect real data without noise. The noise can take various forms: missing values, replaced values, etc. Ideally an interestingness measure must provide "stable" rules even if data is noisy [2].

p12:Intelligibility: The semantic of the measure has a great importance to communicate and explain the results. The measure should be comprehensible by experts and users.

5 Results of the Study

In this section, we establish the first nine semantic properties for the different sixty two measures (table 4). To evaluate the property p1, we compared $m(A \rightarrow B)$ and $m(B \rightarrow A)$. To evaluate the properties p5, p6, p7 and p8, we calculated the values of the sixty two measures in the four reference situations. To evaluate the properties p2, p3, p8 and p9, we draw the corresponding curves.

Table 4. Semantic properties of interestingness measures

| | | measures | | | | | | | | | | | | | | | | |
|------------|----|----------|----|----|----|----|----|----------|----|-----------|----|----|----|----|----|----|----|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | |
| properties | p1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | | |
| | p2 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | |
| | p3 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | p4 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | |
| | p5 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | |
| | p6 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | |
| | p7 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| | p8 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 0 | 0 | 0 | 2 | 1 | 2 | |
| | p9 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | |
| properties | p1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | |
| | p2 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | |
| | p3 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | |
| | p4 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | |
| | p5 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | |
| | p6 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | p7 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | |
| | p8 | 2 | 2 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 2 |
| | p9 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | |
| properties | p1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | p2 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | |
| | p3 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | |
| | p4 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | |
| | p5 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | |
| | p6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | p7 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | |
| | p8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | |
| | p9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | | | |
| properties | p1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | | |
| | p2 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | |
| | p3 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | | | |
| | p4 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | | | |
| | p5 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | | | |
| | p6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | | | |
| | p7 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | | | |
| | p8 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 2 | 2 | 0 | 0 | 1 | 1 | 1 | | | |
| | p9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | |

6 Conclusion and Perspectives

We presented related works which have outlined some semantic properties of some interestingness measures in order to help the user choosing an appropriate one. Our contributions are that: firstly, we identified twelve semantic properties summarizing and completing the studied ones. Secondly, we established the values of nine

semantic properties for sixty two interestingness measures. The evaluation of the intelligibility (p12) is subjective because it depends on the user. The evaluation of the facility to fix a threshold (p10) and sensitivity to the noise (p11) can be done experimentally. We plan to continue the evaluation of these remaining three properties for the sixty two studied measures.

Based on table 4, we can identify some weak measures. The Causal-Support (m25) have not any interesting property. The Support measure (m1), the Rectangular-Gain (m50) and the Collective-Strength (m55) measure have only one interesting property: The sensitivity to incompatibility. The Surprisingness (m43), the Gain (m48), the Quotient (m60) and the Improvement (m61) have only one interesting property: the asymmetry.

We observe also that there is an important measure: The Zhang measure (m7). It has seven interesting properties. It is asymmetric, sensible to A and B , sensitive to independence, logical implication, incompatibility and have a concave shape (linearity property).

As future work, we plan to develop a Multi-criteria decision support system exploiting the results of this study to assist the user choosing an appropriate interestingness measure. We plan also to study the experimental behavior of these interestingness measures in supervised learning (classification task).

References

1. R. Agrawal, T. Imielinski, A. Swami: Mining association rules between sets of items in large databases. ACM SIGMOD Int. Conf. on Management of Data (1993).
2. J. Azé, Extraction de connaissances à partir de données numériques et textuelles. PhD Thesis (2003), University of Paris-Sud, Paris, France.
3. J. Blanchard, Un système de visualisation pour l'extraction, l'évaluation, et l'exploration interactives des règles d'association. PhD Thesis (2005), University of Nantes, France.
4. S. Ben Yahia, Gh. Gasmı, E. Mephu Nguifo and Y. Slimani. A new informative generic base of association rules. In Proceedings of the 2nd Intl. Workshop on Concept Lattices and Applications (CLA'04), Ostrava, Czech Republic, page 67-79, September 2004.
5. C. Borgelt, R. Kruse: Induction of association rules: Apriori implementation. In: 15th Conf on Computational Statistics (2002).
6. H. Cherfi, Y. Toussaint: Adéquation d'indices statistiques à l'interprétation de règles d'association. 6th Int. Conf. On « Analyse statistique des Données Textuelles (JADT) », Saint-Malo, France, March 2002.
7. M.M Gammoudi. Méthode de Décomposition Rectangulaire d'une Relation Binaire : une base formelle et uniforme pour la génération automatique des thesaurus et la recherche documentaire. PhD Thesis (1993), Université Sophia-Antipolis, France.
8. R. Gras, R. Couturier, M. Bernadet, J. Blanchard, H. Briand, F. Guillet, P. Kuntz, R. Lehn, et P. Peter. Quelques critères pour une mesure de qualité de règles d'association - un exemple : l'intensité d'implication. National Journal of Information Technologies (RNTI), France, 2004.
9. F. Guillet, Mesures de la qualité des connaissances en ECD, 2004, Tutorial des Journée Extraction et Gestion des Connaissances (EGC) 2004.
10. T. Fukuda, Y. Moriomolo, S. Morichita and T. Tokuyama. Datamining using two-dimensional association rules: Scheme, algorithms and visualisation. In the ACM-SIGMOD Int. Conf. on the Management of Data. ACM Press, June 1996.

11. R.J. Hilderman, H.J. Hamilton. *Knowledge Discovery and Measures of Interestingness*. Kluwer Academic Publishers, 2001.
12. X-H. Huynh, F. Guillet, and H. Briand (2005). Arqat: an exploratory analysis tool for interestingness measures. *ASMDA'05*, In the 11th Int. Symposium on Applied Stochastic Models and Data Analysis, 2005.
13. Y. Kodratoff. *Comparing Machine Learning and Knowledge Discovery in Databases: An Application to Knowledge Discovery in Texts*. LNAI-Tutorial series, Springer Verlag, 2000.
14. N. Lavrac, P. A. Flach and B. Zupan « Rule evaluation measures: a unifying view », in the ninth Int. Workshop on Inductive Logic Programming, Springer-Verlag, 1999.
15. R. Lehn, F. Guillet, P. Kuntz, H. Briand and Philippé, J. (1999). Felix: An interactive rule mining interface in a KDD process. In P. Lenca, editor, *HCP'99*, pages 169–174.
16. P. Lenca, P. Meyer, B. Vaillant, P. Picouet, S. Lallich. *Evaluation et analyse multicritère des mesures de qualité des règles d'associations*, National Journal of Information Technologies (RNTI), France, pp.219-246, 2004.
17. G. Piatetsky-Shapiro. *Discovery, Analysis, and Presentation of Strong Rules* (chapter 13). In *Knowledge Discovery in Databases*, pages 229–248, AAI/MIT Press, G. Piatetsky-Shapiro and W.J. Frawley edition, 1991.
18. S. Souad-Bensafi, F. LeBourgeois, H. Emptoz, M. Parizeau, *La relaxation probabiliste pour l'étiquetage logique des documents : applications aux tables des matières*, PhD Thesis (2001), Ottawa, Canada.
19. A. Totohasina, H. Ralambondrainy and J. Diatta. *Notes sur les mesures probabilistes de la qualité des règles d'association: Un algorithme efficace d'extraction es règles d'association implicatives*. In the 7th African Conference on Research in Computer Sciences (CARI'04), Hammamet, Tunisia, November 2004.

A New Mutation Operator for the Elitism-Based Compact Genetic Algorithm

Rafael R. Silva, Heitor S. Lopes*, and Carlos R. Erig Lima

Bioinformatics Laboratory, Federal University of Technology Paraná (UTFPR),
Av. 7 de setembro, 3165 80230-901, Curitiba (PR), Brazil
rafael.rsi@gmail.com, hslopes@pesquisador.cnpq.br, erig@utfpr.edu.br

Abstract. A Compact Genetic Algorithm (CGA) is a genetic algorithm specially devised to meet the tight restrictions of hardware-based implementations. We propose a new mutation operator for an elitism-based CGA. The performance of this algorithm, named emCGA, was tested using a set of algebraic functions for optimization. The optimal mutation rate found for high-dimensionality functions is around 0.5%, and the low the dimension of the problem, the less sensitive is emCGA to the mutation rate. The emCGA was compared with other two similar algorithms and demonstrated better tradeoff between quality of solutions and convergence speed. It also achieved such results with smaller population sizes than the other algorithms.

1 Introduction

Since long ago, Genetic Algorithms (GA) have been used as efficient tools for optimization problems, not only in Computer Science, but also in Engineering [4]. For most applications, GAs are implemented in software running on general-purpose processors. However, for applications that require the algorithm to run in real-time, hardware-based implementations are more adequate. Reconfigurable logic can be used for such implementations, by using high-performance FPGA (Field Programmable Gate Array) devices [2]. Even the most advanced devices have limited resources, specially regarding available memory. The Compact Genetic Algorithm (CGA) was devised to meet tight requirements of hardware-based implementations. For instance, a CGA represents a population of individuals by using a single probability vector, thus reducing significantly the amount of memory needed.

The CGA has been sparsely explored in the recent literature. Therefore, there is much room for theoretical development and research that can improve its efficiency. In this work we present a new mutation operator for an elitism-based CGA, capable of better controlling the selective pressure and improving the quality of solutions found. This is achieved without significant decrement of the convergence speed of the algorithm. Following the terminology previously

* This work was partially supported by the Brazilian National Research Council – CNPq, under research grant no. 305720/2004-0 to H.S. Lopes.

used by other authors, the proposed algorithm is named “Elitism with Mutation Compact Genetic Algorithm”, or, in short, emCGA. The performance of the new operator is analyzed by means of computational simulations. We also perform simulations comparing the proposed algorithm with other algorithms proposed in the literature.

This paper is divided as follows. In Sect. 2 a formal description of the CGA is presented, with special focus on those works that use elitism to minimize the perturbations caused by the crossover operator in a CGA. The emCGA is presented in details in Sect. 3. Section 4 presents the results of the computational simulations. Finally, Sect. 5 presents the conclusions of the work.

2 The Compact Genetic Algorithm

The Compact Genetic Algorithm was first proposed by [5]. This algorithm uses a random-walk approach to represent a conventional genetic algorithm in a compact way. This technique is a stochastic process that formalizes successive steps towards a random direction. CGA simulates random-walks for each bit in the chromosome.

In short, in the CGA, individuals are generated randomly based on a probability vector and, at each generation, a tournament among individuals takes place. The probability vector is then updated towards the tournament winner. The elements of this vector represents the probability that each bit in the individual’s chromosome to be either 0 or 1. The population of individuals is, therefore, represented compactly and it converges when all the elements of the probability vector reach either 0% or 100%. Since the probability vector represents itself the whole population, the amount of memory necessary to hold the population is much smaller when compared to a conventional GA. The memory resources needed by a conventional GA are estimated in $N \times L$ bits, where N is the population size and L is the number of bits of the chromosome. For a CGA, the elements of the probability vector are quantized with resolution $1/N$. Therefore, the amount of memory resources needed by a CGA falls down to only $L \times \log_2(N)$. This feature of CGA makes it an appealing alternative for hardware implementations, rather than the conventional GA.

In the CGA, the selection phase is called random generation of chromosomes, or simply, generation, and the reproduction phase is known as updating the probability vector, or simply, updating. In a conventional GA, the Darwinian principle of survival of the fittest is most evidenced in the selection phase, where high-fitted individuals have more chance to survive and spread their genetic material. In a CGA, this phase is embedded in the generation. Recalling that the probability vector is updated towards the direction of the tournament winner at each generation, the chromosome will tend to retain the genetic information of the best individuals, thus influencing the upcoming generations.

Reproduction, the other phase of the algorithm, is typically represented in a conventional GA by the application of genetic operators, mainly crossover and mutation. Crossover is an operator capable of recombining parts of parental

chromosomes and generating two new offsprings. Throughout generations, the repeated applications of crossover leads to a decorrelation of the genes in a population. According to [5], in such decorrelated state, a simple probability vector can be a more compact and suitable representation for the whole population of individuals. In this case, updating the probability vector is analog to the crossover operator action in the conventional GA. Most of the CGAs reported in the literature do not use the mutation operator.

The typical stopping criterion for a conventional GA is when the number of generations achieves a predefined number of generations. In a CGA, the stopping criteria cannot be other than the convergence of the probability vector.

Crossover can generate critical perturbations in problems with high-order building blocks [4]. Harik and colleagues [5] demonstrated that such perturbations can be minimized when the selective pressure is increased. Further, [1] showed that elitism is even more suited for this purpose. This fact has motivated the emergence of several elitism-based CGAs in recent years [1], [3].

When elitism is used in a conventional GA, one or more top-fitted individuals are copied with no change to the next generation. In the original CGA, at each generation, two individuals are randomly generated using the probability vector. These two individuals compete in a tournament. In an elitism-based CGA, only one individual is randomly generated and the other competitor of the tournament is a copy of the best individual of the current generation. Examples of elitism-based CGAs are: persistent elitist CGA (pe-CGA) and nonpersistent elitist CGA (neCGA) [1], and CGA with elitism and mutation (mCGA) [3].

Depending on the nature of the problem dealt by a CGA, the use of elitism can induce a too high selective pressure. Hence, some technique for controlling selective pressure is necessary. Inheritance control of the best individuals was proposed by [1] in the neCGA. Also, a mutation operator was proposed by [3] in the mCGA. Both works present important improvement in the quality of solutions obtained by the algorithm.

Two features of CGA make it appealing for hardware implementations: the binary representation of solutions and its small demand of memory resources. However, a CGA does not explore all the typical features of a conventional GA and, thus, a more limited performance is expected. Therefore, this fact suggests that, by using specific features of a conventional GA in a CGA it is fair to believe that a better performance can be achieved, as shown later in this work.

3 The emCGA

The previously mentioned works (neCGA and mCGA) perform better than the original CGA [1]. This is obtained by minimizing the perturbation provoked by the crossover operator, and thus, leading to a significant improvement in the quality of solutions. In this work we propose a new mutation operator, aimed at improving even more the quality of obtained solutions but, also, keeping a reasonable convergence speed. This operator allows a more efficient control of the selective pressure, adjusting the population diversity as the consequence of

the manipulation of the probability vector. Comparing the proposed mutation operator with that of mCGA [3], the new operator decreases the number of tournaments per generation and, consequently, the total number of fitness evaluations per generation. The consequence is a significant improvement in the convergence speed of the algorithm. On the other hand, the mutation in the mCGA works on the best individual of the current generation. This method requests a new tournament and, consequently, one more fitness evaluation.

The new CGA resulting from the use of the proposed mutation is named emCGA (elitism with mutation CGA). Basically, the proposed mutation operator changes the random generation phase, by modifying the chromosome generated by the probability vector.

4 Computational Experiments and Results

Mathematical functions have been frequently used as a benchmark for optimization algorithms, including GAs. In this work we selected a suite of complex mathematical functions defined in a n -dimensional space (\mathcal{R}^n). This suite includes the following problems, with respective dimensions: Sphere(15), Circle(2), Shaffer-F6(2), Griewank(15), Discrete-1 (15) and Discrete-2(15). Such problems were chosen because some of them were used to evaluate performance of evolutionary computation algorithms [6].

For the Sphere problem (Sect. 4.1) we used a population of 255 individuals, whereas for the remaining problems (Sect. 4.2) we analyzed the performance of the algorithm using 31, 63, 127, 255, 511, 1023 and 2047 individuals. These values were chosen based on the current literature. The stopping criterion is not based on a predefined number of generations, but in the convergence of the probability vector. This approach leads to a variable number of generations, and this in an observable parameter in these experiments.

All functions represent minimization problems, and the optimal solution is a multidimensional null vector. For each problem, a different representation was used for the elements of the vectors applied to the functions. For the Circle problem we used 16 bits to represent the range from -32.767 to 32.768. For Discrete-1 and Discrete-2, we used 16-bit integers, ranging from -32768 to 32767. For the remaining problems, we used 18 bits to represent the range -131.071 to 131.072.

For each problem, 100 independent runs were done. Values reported in Sect. 4.1 and Sect. sec-compare are: the average best fitness value and the average number of generations until the convergence of the probability vector.

4.1 Analysis of the Mutation Parameter in the emCGA

The first group of experiments aimed at finding the best value for the mutation rate in the emCGA. The higher the dimension of the problem, the larger the chromosome size and the more complex the problem becomes. Therefore, this experiment will evaluate how the dimension of the problem affects the best mutation

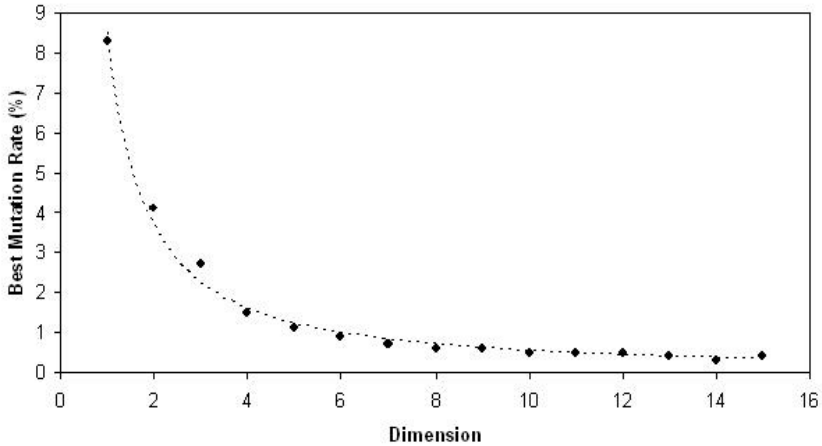


Fig. 1. Best mutation rate as a function of the dimension, for the Sphere problem

rate. For this experiment we used the Sphere problem up to 15 dimensions, and mutation rate ranging between 0 and 15%, with resolution of 0.1%.

In this experiment we observed that the best value for the mutation rate is a function of the dimension of the problem. The best mutation rate is inversely proportional to dimension. That is, the mutation rate that leads to the smallest fitness value depends on the dimension of the problem. It was also observed that exists a range of mutation rates for which the best values of fitness are found, resembling a plateau. This range narrows as the dimension grows. Also, at the extremes of that plateau, the behavior is exponential. As the dimension decreases, the problem becomes less complex and, therefore, it also becomes less sensitive and more tolerant to different values of the mutation rate. Figure 1 shows in a simplified way the behavior of the best mutation rate for all dimensions of the problem tested. It can be observed in this figure a nonlinear relationship between the dimension and the best mutation rate. For high dimensions of the problem, the mutation rate that leads the algorithm to the best performance (regarding fitness values) is around 0.5%. On the other hand, when the dimension of the problem is low, the value for the best mutation rate tends to grow exponentially.

The same experiments done for the Sphere problem were repeated for the other problems. Results obtained (not shown here) were qualitatively equivalent

Table 1. Best mutation rates for the problems tested

| Dimension | Problem | Best mutation rate |
|-----------|--|--------------------|
| 2 | Circle, Shaffer-F6 | 8.5% |
| 15 | Sphere, Griewank, Discrete-1, Discrete-2 | 0.5% |

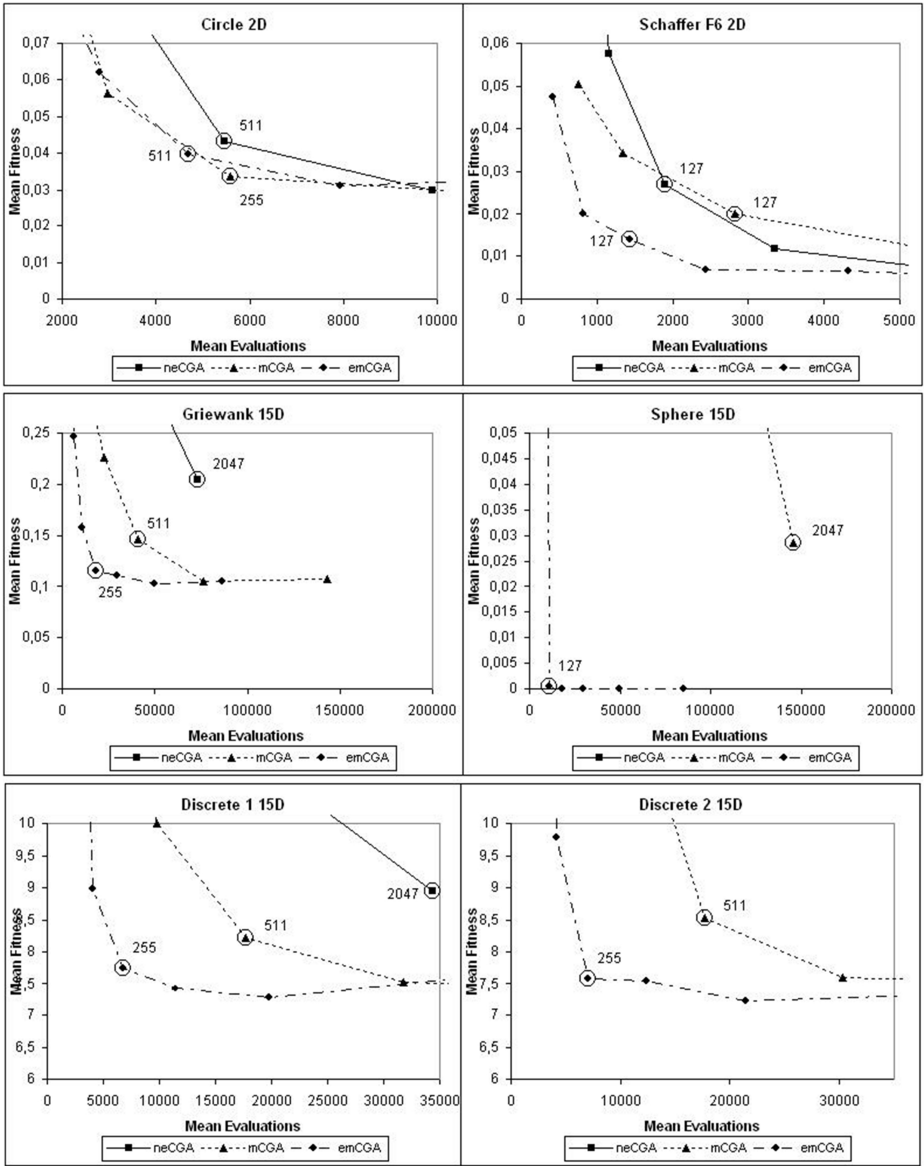


Fig. 2. Comparative analysis in the Pareto front of three CGAs

to those obtained with the Sphere problem. That is, even if we consider different problems, with different dimensions, the behavior is similar to that shown in Fig. 1. This fact suggests that the best mutation rate for a given problem is really dependent on the problem and its dimension. According to these experiments, the best mutation rate found is shown in Table 1.

4.2 Comparison of emCGA with Other Algorithms

Here we compare the performance of the proposed emCGA with other algorithms published in the recent literature, the mCGA [3] and the neCGA [1]. The parameters used in these algorithms are those suggested by respective authors in their publications. In particular, for the mCGA, the mutation rate used was 5% and, for the neCGA the inheritance length was set to 10% of the population size. For the emCGA the mutation rate was set according to the problem, that is, 8.5% for Circle and Shaffer-F6, and 0.5% for the remaining problems.

Considering the nature and purpose of a CGA, for this type of analysis it is important not only the quality of the solution (that is, the best fitness), but also, the number of fitness evaluations to achieve such quality (that is, the number of generations). For a CGA it is important to obtain a good solution in as few as possible generations. Since both objectives are contradictory, the analysis in the Pareto plane can be more useful instead of analyzing both objectives separately. In the Pareto plane, the best tradeoff between the two objectives is that closest to the origin of coordinates system.

Therefore, we used the Pareto plane to compare the behavior of the three algorithms for the test problems (excluding the Sphere problem, used in Sect. 4.1), regarding the best fitness and the number of evaluations. This comparison is shown in Fig. 2. All values in these plots are the average of 100 independent runs. Recall that 100 independent runs were done for each of the several population sizes and for each algorithm. In the plots we show only the results regarding the population size that yielded the best performance for the algorithms. The closest point to the origin is highlighted in the plot with the corresponding population size used. Some of the algorithms had a very poor performance. Consequently its value for fitness and/or number of evaluations was so high that the corresponding point could not fit the scale of the plot.

5 Conclusions

In this work we proposed a new mutation operator for an elitism-based CGA. We analyzed the performance of the proposed operator on several test problems, for different mutation rates and problem dimensions. We also compared the performance of the proposed algorithm with other two recently published algorithms.

The performance analysis of the proposed emCGA reveals that using specific mutation rates for each problem (and each dimensionality) leads to better performance compared with a fixed rate for any problem. Also, we observed a nonlinear relationship between the dimension of the problem and the mutation rate that gives best performance (regarding average fitness). For high-dimensionality problems, a mutation rate of 0.5% seems to be appropriated. For low-dimensionality problems, higher values give better results. However, the low the dimension of the problem, the less sensitive it is to the mutation rate.

Comparing the proposed emCGA with mCGA and neCGA, we observed that, except for the Circle problem, emCGA achieved, at the same time, better

convergence speed and better fitness value than the other algorithms. The Circle problem is the simplest one of the suite, furthermore it is a low-dimensionality problem. Possibly, these are the reasons why emCGA achieved a performance quite similar to mCGA, however with a better convergence speed.

For the high-dimensionality problems (and also for Shaffer-F6 problem), emCGA found solutions using smaller populations than those used by the other algorithms. This is an important issue since such algorithms were proposed for hardware implementations, where memory resources are limited.

Figure 2 shows that emCGA is the algorithm that has the better tradeoff between quality of solution and convergence speed. This fact suggests that emCGA is an interesting alternative for implementations that require compact genetic algorithms.

Future work will focus on evaluating the overall limits of the proposed approach, by implementing emCGA in a FPGA device do deal with a real-world difficult problem such as [7], [8].

References

1. Ahn, C., Ramakrishna, R.: Elitism-based compact genetic algorithms. *IEEE Trans. Evolutionary Computation* **7** (2003) 367–385.
2. Becker, J., Hartenstein, R.: Configware and morphware going mainstream. *J. Systems Architecture* **49** (2003) 127–142.
3. Gallagher, J., Vignanam, S., Kramer, G.: A family of compact genetic algorithms for intrinsic evolvable hardware. *IEEE Trans. Evolutionary Computation* **8** (2004) 111–126.
4. Goldberg, D.: *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley, Reading (1989).
5. Harik, G., Lobo, F., Goldberg, D.: The compact genetic algorithm. In: *Proc. IEEE Conf. on Evolutionary Computation*, (1998) pp. 523–528.
6. Krink, T., Filipic, B., Fogel, G., Thompsen, R.: Noisy optimization problems – a particular challenge for differential evolution ? In: *Proc. IEEE Conf. on Evolutionary Computation*, (2004) pp. 332–339.
7. Lopes, H.S., Moritz, G.L.: A graph-based genetic algorithm for the multiple sequence alignment problem. *Lecture Notes in Artificial Intelligence* **4029** 420–429.
8. Moritz, G.L., Jory, C., Lopes, H.S., Erig Lima, C.R.: Implementation of a parallel algorithm for pairwise alignment using reconfigurable computing. *Proc. IEEE Int. Conf. on Reconfigurable Computing and FPGAs*, (2006) pp. 99–105.

Genetic Programming for Proactive Aggregation Protocols

Thomas Weise, Kurt Geihs, and Philipp A. Baer

University of Kassel, Wilhelmshöher Allee 73, D-34121 Kassel, Germany
{weise,geihs,baer}@vs.uni-kassel.de
<http://www.vs.uni-kassel.de>

Abstract. We present an approach for automated generation of proactive aggregation protocols using Genetic Programming. First a short introduction into aggregation and proactive protocols is given. We then show how proactive aggregation protocols can be specified abstractly. To be able to use Genetic Programming to derive such protocol specifications, we describe a simulation based fitness assignment method. We have applied our approach successfully to the derivation of aggregation protocols. Experimental results are presented that were obtained using our own Distributed Genetic Programming Framework. The results are very encouraging and demonstrate clearly the utility of our approach.

1 Introduction

Determine the highest temperature measured by a sensor in a given area. Find the average load of all computers in a grid. Aggregation functions with their ability to summarize information in a certain, user-specified way are a very important building block for distributed applications [1]. As a standard service of databases, SQL-queries allow the user to aggregate locally available data in one or multiple tables. Performing aggregation in a sensor network [2], as done in the introductory examples, however is more complicated. Only data obtained from the local sensors is available on a node. In order to determine the desired aggregate, a node needs to exchange messages with other nodes in its neighborhood. Therefore, reactive and proactive protocols can be distinguished. Reactive protocols are used to compute queries issued by a single node and offer the result of the query to this node only. Proactive protocols update aggregation values continuously and make them globally available.

In this paper we demonstrate that Genetic Programming is an effective technique to derive proactive aggregation protocols. We describe how such protocols can be specified in an abstract manner and introduce a simulation-based fitness assignment method used to evaluate these specifications.

2 Related Work

In our work we strongly refer to Jelasily et al. who have defined and evaluated many proactive aggregation protocols for large-scale overlay networks [3].

Although their communication model is gossip-based, our abstract protocol specifications will work with epidemic [4] or SPIN-based [5] communication as well.

Protocol generation has been a field of application for Genetic Algorithms in the past ten years. Genetic Algorithms have been used to optimize different aspects of protocols like communication, implementation costs, and performance [6,7,8,9]. While these approaches separate the application logic itself from the protocol generation and focus on the communication, we introduced in our previous work [10] a versatile Genetic Programming technique for simple distributed algorithms combining both of these aspects. In this paper we shift the emphasis towards the application logic (i.e. the formulas needed to compute aggregation values) while using an implicit communication pattern.

As we will show in the next sections, this mathematical logic is based on a form of symbolic regression [11,12] but exceeds its scope in many ways.

3 Basic Model and Protocol Specification

For our studies we use a system model similar to the one introduced by Jelasity et al. [3]. A distributed system is regarded as a large collection of nodes that communicate through message exchange. These nodes are assumed to be connected by a routed network allowing all nodes to exchange messages with each other.

In order to compute an aggregate value (see *target* in Figure 1), each node owns a local storage, consisting of n variables $v_i : i \in 1..n$ named a, b, c, \dots and so on. It furthermore knows one value, for example a sensor measurement, needed to compute the aggregate. With this *provided* value, $0 < m \leq n$ of the variables will be initialized while the others have constant numerical *initial* values.

To perform the aggregation, data exchange and variable updates are needed to be carried out iteratively in a loop. The data exchange in the network is performed as follows: $0 < p < n$ of the variables are marked as *output* and also p variables are marked as *input*. At the beginning of each iteration step, a partner node y is selected for each node x in the network. This could, for example, be a node in transmission range chosen from the direct neighborhood. The values of the output variables of node x will then be stored in the input variables of node y , and vice versa. After this is done, the variables are updated by computing formulas $f_j : j = 1..q$ on every node, where each formula f_j assigns a new value to one of the variables v_i . Formulas are expressions built with operators such as $+$, $-$, $*$, $/$, \wedge (power), $|*|$ (absolute value), \min , \max , constants, and with variables.

Figure 1 displays such a protocol specification, able to compute the aggregate *Average*. This optimal protocol, which was found using our own Distributed Genetic Programming Framework [13,14,15], is exactly the same as the solution proposed by Jelasity et al. [3].

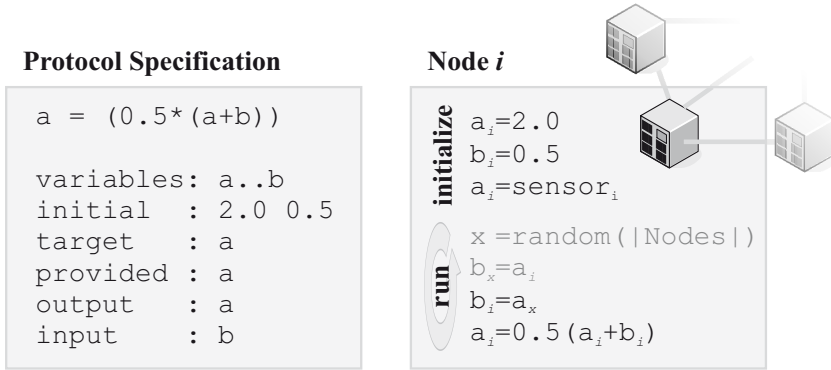


Fig. 1. A protocol specification for the distributed average, found using GP

4 The Fitness Assignment Method

In principle, the evolution of protocol specifications is a complex extension of symbolic regression [11]. Instead of trying to evaluate a single formula with a single input parameter one time for each test set [12], multiple formulas with multiple inputs are evaluated T times in a loop on multiple nodes in a simulated network.

When performing symbolic regression, the functionality of a formula can be determined by computing its result for a number of test sets. The difference between the values computed and the expected results then clearly indicates the fitness.

In case of engineering an aggregation protocol P , each test set is an n -dimensional vector where n is the number of simulated nodes. Running the protocol yields an aggregate value a_y for each one of the nodes ($y = 1..n$) in a network. To decrease the inaccuracy as well as to reward protocols with at least one good approximation, we compute the sum δ of the average and the maximum deviation of these aggregates from the correct result A . As shown in equation 1, $\delta_x(t)$ is the deviation sum of the aggregate values a_y computed by the nodes y for a single test set x . This value should be minimized.

$$\delta_x(t) = \max_{\forall \text{ nodes } y} \{|a_y(t) - A_x|\} + \text{avg}_{\forall \text{ nodes } y} \{|a_y(t) - A_x|\} \quad (1)$$

Instead of calculating this value at the end of each simulation, we integrate it for each of the T iterations of the protocols loop (equation 2). Therefore we do not only measure protocol accuracy but also convergence speed: If a protocol converges faster than another one of the same accuracy, the deviation sums will begin to shrink sooner.

This approach has a minor drawback: Protocols which simply “guess” a constant value for the aggregates may be rewarded with a better fitness (smaller δ values) in most of the iteration steps – if they guess well. Weighting the last

iteration with the square root of the total iteration count T has proven to be a useful countermeasure to prevent this phenomenon.

The formula for the functional fitness f of the protocol P computed using all test sets x is presented in equation 2. This value is subject to minimization.

$$f(P) = \sum_{\forall \text{ testsets } x} \frac{1}{|A_x|} \frac{1}{|T + \sqrt{T}|} \left[\sum_1^{T-1} \delta_x(t) + \sqrt{T} \delta_x(T) \right] \quad (2)$$

Other quality aspects of an aggregation protocol are the size of the messages exchanged, the number of variables needed and the complexity of the formula expressions. Introducing a second fitness function, we use a weighted multi-objective approach to optimize these aspects too.

5 Example

Trivial basic aggregation functions like *Minimum*, *Maximum* and *Average* can be evolved by using the fitness assignment method of the previous section within only a few generations using Genetic Algorithms with less than 2000 individuals. To prove that our approach is also suitable for more complex aggregates, we chose the following function to be approximated: Each node owns one sensor producing one measured value. We are interested in the average of the square roots of the absolute values of these measurements. The most trivial solution would be that the nodes compute initially the square roots of their sensor measures and then execute the *Average* protocol of figure 1. In a real application, the sensor measurements will change over time forcing the protocol to incorporate new values (see figure 2). This is not possible with the trivial solution discussed, which therefore is not a valid protocol specification.

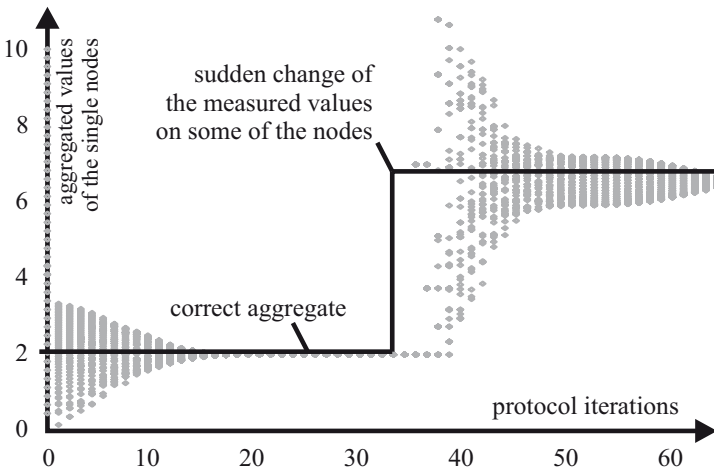


Fig. 2. A Protocol sensitive to input data change

Protocol Specification

```

b = (b/a)
a = |c|
c = max((a*b), (a-(b*a)))
a = ((a*b)^0.5000000022)

```

```

variables: a..c
initial   : 3.14 0.11 154
target    : a
provided  : c
output    : a
input     : b

```

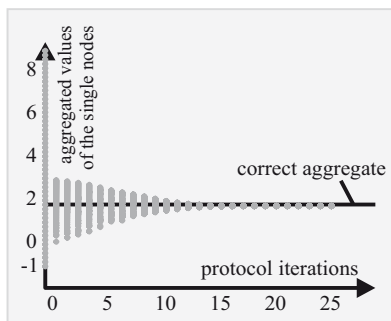
Convergence Diagram

Fig. 3. A protocol evolved for the example problem and its convergence behavior

Since the search algorithm implementations of the Distributed Genetic Programming Framework, which we use for our studies, are intended for maximization only, the functional fitness (see equation 2) was inverted. A Genetic Algorithm with a bigger population of 8192 individuals was applied, yielding the solution depicted in figure 3. This figure shows the functionally best pareto-optimal protocol, which produces approximation values differing only by a very small error from the correct aggregate. The diagram in the right part of figure 3 shows the convergence of the protocol. 50 nodes were initialized with values uniformly distributed over the interval $[-1, 9]$ and quickly converged to a constant shared by all nodes, only deviating less than 1% from the real solution. These deviations rise considerably if the measured input values change, as mentioned at the beginning of this section. The protocol is however able to adapt to bigger changes – it is also the source of figure 2.

It is important to mention here that genetically evolved protocol specifications have a strong affinity to memorize test sets. Symbolic regression often does not yield the “original” formula which was used to create the test sets but a function which only resembles this formula closely. In symbolic regression this effect is wanted in many cases – for protocol generation it is devastating. Sometimes generated protocols which seem to be excellent solutions for a given aggregation problem turn out to be disguised decision tables. In our work we use two simple means to circumvent this problem:

1. We use many test sets (empirically determined ≥ 240).
2. The values contained in the test sets differ in scale, sign, and generating distribution function (uniform, normal, many zeros/few ones).

Another interesting phenomenon concerns the fitness of the protocol specifications. Plotting the maximum and the median of the functional fitness of each generation exhibits an interesting behavior (see figure 4): while the fitness of the best individuals becomes significantly better step by step, the fitness of most of the population degenerates at the same rate. A possible explanation for this phenomenon would be that the more accurate protocol specifications get, the

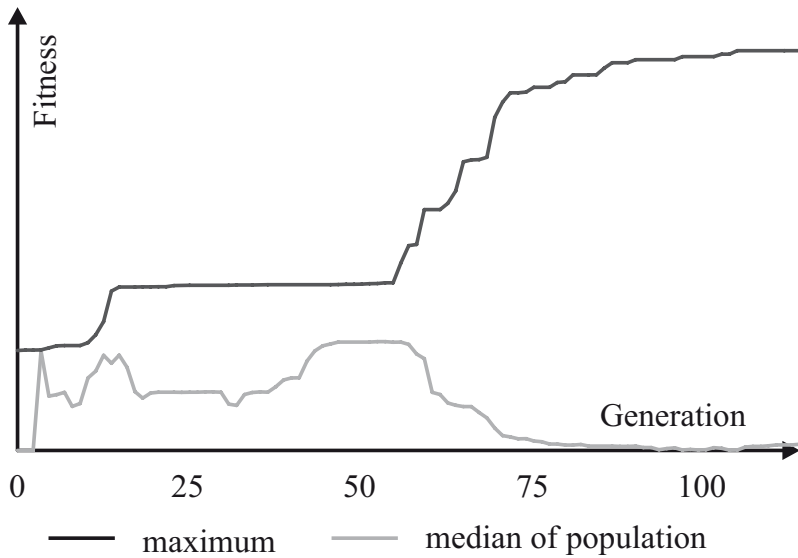


Fig. 4. The maximum and median fitness of the whole population of the Genetic Algorithm for the average-of-square-roots problem, plotted against the generations

more destruction can be done by genetic operators. This effect is increased by the non-functional fitness function applied which adds pressure towards compact protocol design – and therefore tends to shrink those parts of the specification that have smaller impact on the total results.

6 Conclusion and Future Work

In our research we were able to demonstrate the utility of Genetic Programming for deriving proactive aggregation protocols. Thus, after having shown its usefulness for breeding assembler-like distributed algorithms in [14], in this paper we have presented our second successful application of GP to create emergent phenomena: A global property of a whole (network) has been transformed into behavior rules for an individual (node). In our future work we will go on evaluating different fields of application of GP as a means for creating distributed algorithms. Our DGPF [13] will be improved further with the ultimate goal to provide a comprehensive platform for automated software creation for sensor networks.

References

1. Robbert van Renesse. The importance of aggregation. (2584):87–92, 2003.
2. Chee-Yee Chong and S.P. Kumar. Sensor networks: evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, Aug 2003.
3. Márk Jelasity, Alberto Montresor, and Ozalp Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Trans. Comput. Syst.*, 23(1):219–252, 2005.

4. Márk Jelasity and Alberto Montresor. Epidemic-style proactive aggregation in large overlay networks. In *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, pages 102–109, Tokyo, Japan, Mar 2004. IEEE Computer Society.
5. Wendi Rabiner Heinzelman, Joanna Kulik, and Hari Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 174–185, New York, NY, USA, 1999. ACM Press.
6. K. El-Fakih, H. Yamaguchi, G. Bochmann, and T. Higashino. A method and a genetic algorithm for deriving protocols for distributed applications with minimum communication cost. In *Proceedings of Eleventh IASTED International Conference on Parallel and Distributed Computing and Systems*, Nov 1999.
7. Lidia Yamamoto and Christian Tschudin. Genetic evolution of protocol implementations and configurations. In *IFIP/IEEE International workshop on Self-Managed Systems and Services (SelfMan 2005)*, 2005.
8. F. Comellas and G. Giménez. Genetic programming to design communication algorithms for parallel architectures. *Parallel Processing Letters*, 8(4):549–560, 1998.
9. Marcio Nunes de Miranda, Ricardo N. B. Lima, Aloysio C. P. Pedroza, and Antonio C. de Mesquita. HW/SW codesign of protocols based on performance optimization using genetic algorithms. Technical report, 2001.
10. Thomas Weise and Kurt Geihs. DGPF - an adaptable framework for distributed multi-objective search algorithms applied to the genetic programming of sensor networks. In Jurij Šilc Bogdan Filipič, editor, *Proceedings of the Second International Conference on Bioinspired Optimization Methods and their Application, BIOMA 2006*, International Conference on Bioinspired Optimization Methods and their Application (BIOMA), pages 157–166. Jožef Stefan Institute, Ljubljana, Slovenia, Oct 2006.
11. John R. Koza. *Genetic Programming, On the Programming of Computers by Means of Natural Selection*. A Bradford Book, The MIT Press, Cambridge, Massachusetts, 1992 first edition, 1993 second edition, 1992.
12. Gunther R. Raidl. A hybrid GP approach for numerically robust symbolic regression. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David E. Goldberg, Hitoshi Iba, and Rick Riolo, editors, *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 323–328, University of Wisconsin, Madison, Wisconsin, USA, 22–25 1998. Morgan Kaufmann.
13. Distributed Genetic Programming Framework. SourceForge project, see <http://sourceforge.net/projects/DGPF> and <http://DGPF.sourceforge.net/>.
14. Kurt Geihs Thomas Weise. Genetic programming techniques for sensor networks. In *Proceedings of 5. GI/ITG KuVS Fachgespräch "Drahtlose Sensornetze"*, pages 21–25, Jul 2006.
15. Thomas Weise. Genetic programming for sensor networks. Technical report, Jan 2006.

Automatic Synthesis for Quantum Circuits Using Genetic Algorithms

Cristian Ruican, Mihai Udrescu, Lucian Prodan, and Mircea Vladutiu

Advanced Computing Systems and Architectures Laboratory
University Politehnica Timisoara, 2 V. Parvan Blvd., Timisoara 300223, Romania
{crys,mudrescu,lprodan,mvlad}@cs.upt.ro
<http://www.acsa.upt.ro>

Abstract. This paper proposes an automated quantum circuit synthesis approach, using a genetic algorithm. We consider the circuit as a successive rippling of the so-called gate sections; also, the usage of a database is proposed in order to specify the gates that will be used in the synthesis process. Details are presented for an appropriate comparison with previous approaches, along with experimental results that prove the convergence and the effectiveness of the algorithm.

1 Introduction

1.1 Quantum-Inspired Genetic Algorithm

The field of Evolvable Quantum Information (EQI) has significantly grown over the last years [7]. At first glance, the merging of quantum computation and evolvable computation seems natural and benefic. Indeed, relevant progress has been signaled in the EQI subfield of Quantum-Inspired Genetic Algorithms (QIGA) including the so-called evolvable quantum hardware or the automatic synthesis of quantum circuits by evolvable means [7]. Ongoing developments concerning the other EQI subfield of Quantum Genetic Algorithms (QGA) have been presented previously [8].

This paper concerns the QIGA field, and attempts to automatically build quantum circuits that implement a given unitary transformation (i.e. automated quantum circuit synthesis) by means of Genetic Programming. Our motivation was to evolve complex quantum circuits.

1.2 Background

In quantum computation the qubit is the basic unit of information. In Bra-Ket notation, a qubit is a normalized vector in a two dimensional Hilbert space $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, $|\alpha|^2 + |\beta|^2 = 1$ ($\alpha, \beta \in \mathbb{C}$), where $|0\rangle$ and $|1\rangle$ are the basis states [6]. The quantum system is described by a superposition of the basis states whereas a classical binary system can only settle in one of the basis states '0' or '1' [2]. The qubits can be organized in linear structures called quantum registers, encoding a superposition of all possible states of the classical register. For a n -qubit quantum

register, its corresponding state is a normalized vector in a \mathcal{H}^{2^n} space, $|\psi_n\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$, where $\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1, i \in \mathbb{N}$. Quantum circuits are constrained networks of gates with no cloning and no feedback allowed [6]. The quantum gate is a physical device implementing an unitary operator that represents the quantum state transformation. Due to the unitary property, all quantum circuits are reversible and are considered the most feasible implementation for quantum algorithms.

Figure 1 presents several quantum gates [1] used by our algorithm’s database.

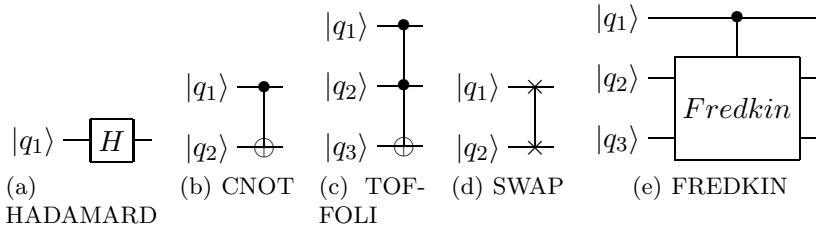


Fig. 1. Elementary gates

1.3 Proposed Approach

We present a new approach that facilitates the successful synthesis for different quantum circuit types that are described with real number matrix elements. The desired output function is provided to the synthesizer, and the tool computes whether a quantum circuit (composed of one or more quantum gates from our database), that implements the function, exists.

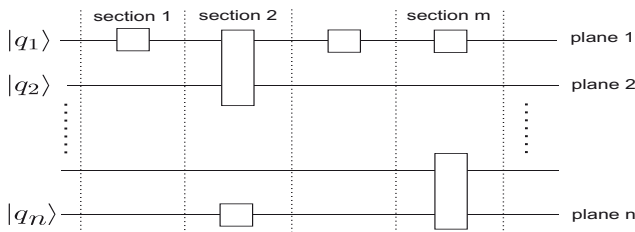


Fig. 2. The quantum circuit view that is used for the new approach

The novelty of the approach lies in splitting the potential circuits in vertical levels called “sections” and horizontal levels called “planes” (Fig. 2), for a consistent representation of the genetic algorithm, which is used for the chromosome definition. Information is exchanged from left to right with the upper wires representing the most significant qubits.

Potential circuits are represented by individuals (Fig. 3(b)), each containing rows for possible gates on which the tensor product can be applied, and one column for the section decomposition. The number of gates on a row is limited by the number of circuit qubits. The number of gate elements within the column is not limited and the level of decomposition can be set interactively. Each individual represents a possible solution, which is computed by applying the tensor product for all horizontal rows and then multiplying all the results.

2 Previous Work

As proposed previously [3], a four-phase design flow assists computations by transforming a quantum algorithm from a high-level language program into precisely scheduled physical actions. The first three phases are implemented by a compiler and the last phase is hardware dependent (in some cases can be just a simulator). The simulation and layout tools incorporate details of the emerging quantum technologies that would ultimately implement the algorithms described by higher level languages.

In reference [10], Lukac and Perkowski have identified the following question: how the number of wires and the gate position within the circuit be encoded by employing the least complex data structure? They proposed a transformation of the quantum circuit in an encoded chromosome, in order to be used in a standard genetic algorithm. In the encoded chromosome the following rules are imposed: equal probability of presence of each gate type, fast encoding and decoding of an individual and no other parameters beside basic definitions (no control bits). The potential weak point is that beside the gate order, there is no information indicating what gates are connected to what wires. Thus, in order to obtain the chromosome, it is required that the quantum circuit be altered by employing swap gates.

Rubinstein, in reference [11], considers for the genetic algorithm a scheme in which a gate has a type, a number of sets for the qubit operands and some sets of parameters for different categories (the generalised 2-qubit gate takes four real parameters for different types of rotations; the CNOT gate takes a number of control qubits, etc). The quantum circuit is considered as a list of gate structures, where the size of the circuit (number of gates) is variable.

An application of a genetic algorithm for evolving quantum computing circuits has also been proposed in reference [4]. The genetic algorithm automatically searches for the appropriate circuit design that yields the desired output state. The fitness function compares the current output with the desired output, the search being stopped when a close match is found.

Shende et al. proposed a top-down structure and effective computation by employing the Cosine-Sine Decomposition [5]. With the help of an optimized quantum multiplexor, a quantum analog Shannon decomposition of Boolean functions is derived, by applying this decomposition recursively to quantum operators. This leads to a circuit synthesis algorithm in terms of quantum multiplexors.

3 A New Genetic Algorithm Approach

3.1 Implementation of Quantum Gates

The algorithm proposed in this paper operates with a collection of quantum gates, that are correspondingly encoded as matrices. A dynamic approach for the matrix representation (list-in-a-list) was used. This approach allowed for efficient memory usage due to different gate types used for synthesis. In order to decrease the time required by the Kronecker product and multiplication, we have created several dedicated memory locations for intermediate results storage.

3.2 The Genetic Algorithm

The special purpose algorithm [9] is used for the automated search of the best quantum gate composition, in order to implement a given quantum multi-qubit unitary transformation.

Our main focus was to define all the phases required by the algorithm, allowing interactive specification of a minimal set of parameters, mainly those necessary for defining the exit point (algorithm stop condition) and those for increasing or decreasing the solution convergence (depending on mutation/crossover percentage, multiple mutation/crossover percentage). Figure 3(a) presents our genetic algorithm. An important step in defining the fitness function (as percentage derived from the matrix comparison) was to find the best encoding for the chromosome (plane and section decomposition) by taking into account, at the same time, the matrix dynamic representation. For example, we compute in parallel the tensorial product for one individual on each section, and only after that we perform the multiplication between sections. This parallel execution of operations reduce the general computation time.

3.3 The Implementation

When encoding the chromosomes (Fig. 3(b)), each individual will contain one or more sections, each section containing one or more gates. The number of gates is given by the number of qubits in the circuit, which is computed from the number of matrix elements (i.e. number of qubits = \log_2 (number of elements)). The number of sections is specified interactively and represents the granularity level of synthesis (a higher number of sections means that the circuit will be composed of elementary gates, and a small number of sections means that the algorithm will search in order to find more complex gates for synthesis). Depending on this parameter, we may find a solution.

The approach randomly selects an individual, then picks a random gate solution. This gate solution will suffer a mutation, thus being replaced by a new randomly generated gate (all gates are defined in the input database). Because we allow multiple mutations, the same individual may suffer another mutation within the defined probability (Fig. 4(a)). This operator is very important when searching the problem space, in order to avoid being trapped in a local minimum of the fitness function.

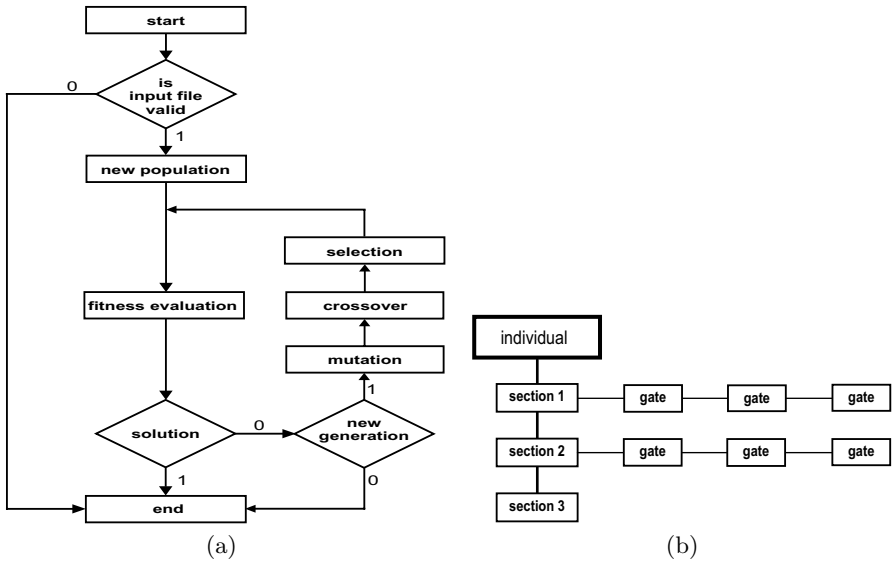


Fig. 3. Genetic algorithm (a) and chromosome encoding (b)

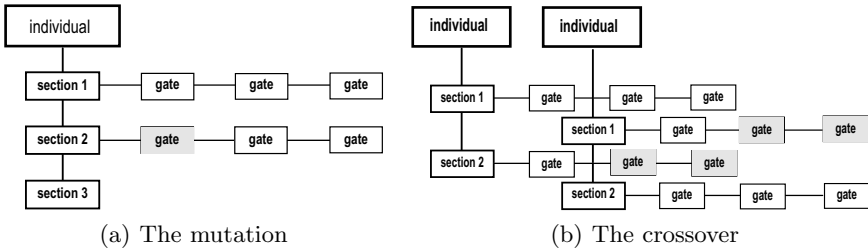


Fig. 4. Genetic operators

Also, we implemented a one-point crossover (Fig. 4(b)): two parents are randomly selected together with a point on their sections, and from that point all the gates between are swapped. The crossover results are used to improve the fitness value of the selected chromosome, allowing the exploration of a larger problem space.

When implementing selection it is important to eliminate the individuals from the generation that manifest a small solution convergence. We used a proportional selection, so that an individual with a fitness value that is smaller than a threshold will be eliminated from the population, and a new one will be created randomly (intruder in population).

We defined our fitness function as a matching percentage with the given output function by comparing each matrix element from our chromosome with each

corresponding element from the expected solution. For instance if our chromosome has three out of four elements that are identical with the given output, the fitness function will be 0.75 .

4 Experimental Results

4.1 Two-Qubit Circuit

Suppose we have the following unitary transformation (1) as an output function:

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \tag{1}$$

and the algorithm is initialized with the default values, a solution will emerge immediately (Fig. 5(a)). If we repeat the algorithm, a second solution will emerge

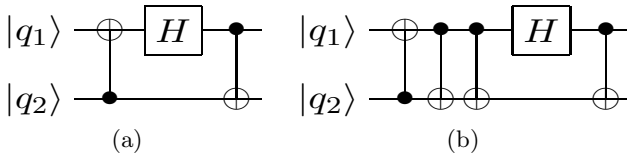


Fig. 5. Circuit synthesis of equation (1)

implementing the same output function (Fig. 5(b)); however it is no longer an optimal solution.

4.2 Three-Qubit Circuit

Suppose we have the unitary transformation (2) as an output function:

$$\frac{1}{2} \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2}$$

two solutions may be obtained by running our genetic algorithm, as shown in Fig. 6.

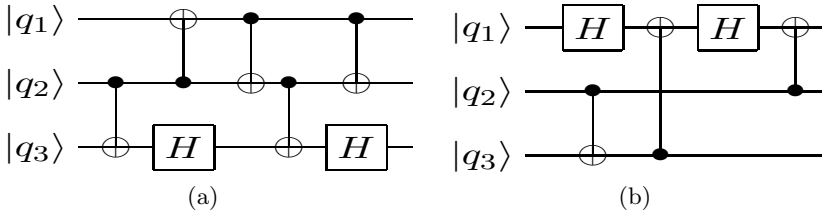


Fig. 6. Circuit synthesis of equation (2)

4.3 Multi-Qubit Circuit

The algorithm successfully generates solutions for multi-qubit circuits (i.e. we emerged solutions for a 5-qubit circuit).

4.4 Benchmark

We used the benchmark proposed in reference [10] to compare our experimental results. Thus, we performed the following tests in the imposed conditions: one-point crossover, mutation can erase, add, increase or decrease the chromosome length, the number of individuals is relatively small (50-100), the maximum number of generations is 50-100, the mutation probability is 0.2-0.6 and the crossover probability is 0.2-0.6. We used gates with 1, 2, 3 and 4 wires (see Table 1) with real elements. The tests were performed for 20 runs in total, the average result being used for comparison.

Table 1. Gates used in benchmark

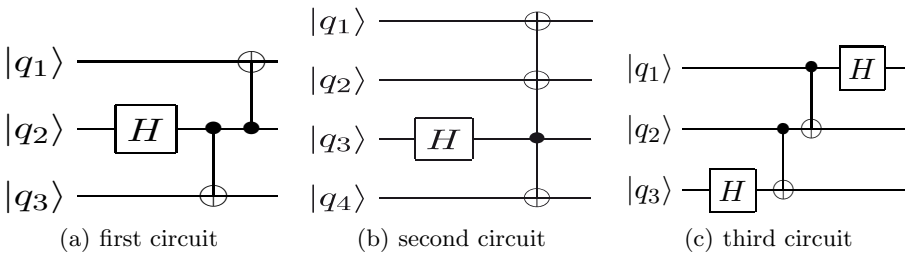
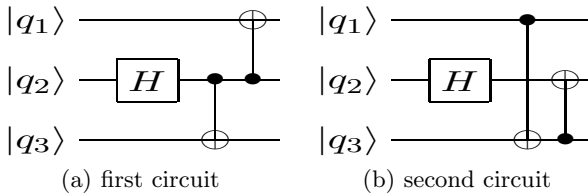
| Number of inputs | Gates |
|------------------|------------------------------|
| 1 | Wire, Hadamard, Pauli (X, Z) |
| 2 | CNOT, Swap, Controlled-Z |
| 3 | Toffoli, Fredkin |

The first test, described in Table 2 is intended to prove the algorithm convergence. All the gates are used with the defined scope of discovering similar circuits. Each gate is used as target gate for the genetic algorithm and the expected result shall be the same gate, a smaller one or a circuit having the same function.

Our approach resulted in better runtime results compared with [10]. Moreover, if multiple mutations and multiple crossover parameters are used, the algorithm becomes convergent in a smaller time, because of the increasing number of possible gate recombination. We may conclude that there is a link between mutation and the search time; a small search time will imply a high number of mutations and few mutations will imply an increased search time. For small gate situations (with one or two inputs), the convergence is fast due to recombination restrictions (i.e. a three input circuit cannot be used to perform synthesis for a two

Table 2. Test the convergence of our approach

| Number of inputs per q-gate | pM | pC | Population size | Number of generations | Real time (average 20 runs) as in [10] | Runtime (average 20 runs) |
|-----------------------------|-----|-----|-----------------|-----------------------|--|---------------------------|
| 1-input | 0.4 | 0.6 | 50 | <50 | <30sec | <0.1sec |
| 1-input | 0.2 | 0.6 | 50 | <50 | <60sec | <0.1sec |
| 2-inputs | 0.6 | 0.4 | 50 | <50 | <30sec | <1sec |
| 2-inputs | 0.2 | 0.4 | 50 | <50 | <60sec | <1sec |
| 3-inputs | 0.6 | 0.6 | 50 | <50 | <60sec | <3sec |
| 3-inputs | 0.2 | 0.6 | 50 | <50 | <180sec | <5sec |

**Fig. 7.** Synthesis of composite circuits**Fig. 8.** Circuit synthesis from [11]

input circuit). The time increase depends on the number of inputs used by the search circuit.

The second test is intended to prove the effectiveness of the synthesis of composite quantum circuits. In [10], three circuits are used, as described in Fig. 7, but the starting set of the used gates was opened. Without having the same prerequisites, we cannot compare our approaches. Using different input gates, we were able to emerge solutions for all of them in a less time than 60 seconds, which is the minimum time obtained in [10] for this specific test.

In reference [11], Rubinstein proposes two three-qubit circuits (see Fig. 8), for the entangled qubit production problem and presents his experimental results.

Table 3. EPR problem synthesis

| pM | pC | Population size | Number of generations | Runtime as in [11] | Runtime |
|------|-----|-----------------|-----------------------|-------------------------------|---------|
| 0.01 | 0.8 | 5000 | 50 | between 30sec and few minutes | <4sec |

We derived, from our approach, better runtime results using the same quantum gate set and the same values for the genetic algorithm, as is presented in Table 3.

5 Conclusion

In the quest for automated quantum circuit synthesis, the approach presented herein, which uses genetic algorithms, has produced different solutions by interactively changing the parameters of the genetic algorithm, or by just repeating the execution.

The resources employed by the algorithm (memory and CPU time) increase with the complexity of the output function. This is due to the number of qubits needed by the algorithm and because of the mathematical operations involved (i.e. the tensor product).

The algorithm will find circuits that implement a given unitary transformation (i.e. the output functions). The tool uses, as input, a database that can be updated interactively with new unitary matrixes representing new quantum circuits.

The major difference, with respect to other approaches is the way circuits are seen as being split in sections and planes, in order to facilitate a better chromosome definition. The optimal encoding of the chromosome allows for a low complexity synthesis of the quantum circuit. Therefore, our proposed genetic methodology offers, at least in the considered experiments, better performance in terms of time; for instance the methodology described in [4] is working only on two qubits.

Dynamical memory allocation allows for efficient time and memory consumption, as well as for the synthesis of different types of circuits without any prior parametrization.

Our motivation was to evolve more complex quantum circuits, and not the benchmark evaluation of the convergence and the effectiveness of the genetic algorithm, although the presented results seem to outperform those obtained with the previous approaches.

Future work will focus on chromosome encoding through graph representation (m -graphs may be suitable), along with an enhanced matrix representation.

References

1. Ekert, A., Hayden, P., Inamori, H.: Basic Concepts in Quantum Computation. Centre for Quantum Computation, University of Oxford. (2000)
2. Malossini, A., Blanzieri, E., Calarco, T.: QGA, A Quantum Genetic Algorithm. Technical Report DIT-04-105. (2004)

3. Svore, K., Cross, A., Aho, A., Chuang, I., Markov, I.: Toward a Software Architecture for Quantum Computing Design Tools. *IEEE Computer*. (2006)
4. Singh, P.: Evolving Quantum Circuits using Genetic Algorithm. *Quantum Physics*, quant-ph/0511036. (2005)
5. Shende, V.V., Bullock, S.S., Markov, I.L.: Synthesis of Quantum Logic Circuits. *Quantum Physics*, quant-ph/0406176. (2004)
6. Nielsen, M., Chuang, I.: *Quantum Computation and Quantum Information*. Cambridge University Press. (2000)
7. Spector, L.: *Automatic Quantum Computer Programming. A Genetic Programming Approach*. Kluwer Academic Publishers. (2004)
8. Udrescu, M., Prodan, L., Vladutiu, M.: Implementing Quantum Genetic Algorithms: A Solution Based on Grover's Algorithm. *Proceedings of the 3rd Conference on Computing Frontiers*. (2006) 71–82
9. Ruican, C.: QSin - A Quantum Circuit Synthesis Software. Download version available on http://www.cs.upt.ro/~crys/index_files/public/qsin.zip. (2006)
10. Lukac, M., Perkowski, M.: Evolving quantum circuits using genetic algorithm. *Proceedings of the 2002 NASA/DOD Conference on Evolvable Hardware*. (2002)
11. Rubinstein, B.I.P.: Evolving quantum circuits using genetic programming. *Proceedings of the 2001 Congress on Evolutionary Computation*. (2001)

Clonal Selection Approach with Mutations Based on Symmetric α -Stable Distributions for Non-stationary Optimization Tasks

Krzysztof Trojanowski

Institute of Computer Science, Polish Academy of Sciences
Ordona 21, 01-237 Warsaw, Poland
trojanow@ipipan.waw.pl

Abstract. Efficiency of two mutation operators applied in a clonal selection based optimization algorithm AIIA for non-stationary tasks is investigated. In both operators traditional Gaussian random number generator was exchanged by α -stable random number generator and thus α became one of the parameters of the algorithm. Obtained results showed that appropriate tuning of the α parameter allows to outperform the results of algorithms with the traditional operators.

1 Introduction

Clonal selection based algorithms belong to the group of evolutionary approaches to optimization. They were inspired by the clonal selection phenomenon which is present in the human immune system. The phenomenon is responsible for the ability of adaptation to new patterns of invading organisms. It is a part of the so called primary immune response of the system. The clonal selection paradigm was an inspiration for a set of heuristic approaches to optimization since the end of 90's [12].

In our research we investigate the use of the clonal selection principle based approaches for non-stationary optimization tasks. In our previous research [9] it was shown that some of them are more successful than the others in optimization of a suite of tasks created with two known test-case generators [2,6,10]. Thus for our recent investigations the AIIA algorithm was selected as the most efficient of them. The results of experiments with two types of mutation operator are presented in this paper. Our goal was to study the influence of the application of random number generators with different types of distributions over the efficiency of these operators.

In Section 2 the AIIA algorithm is briefly reminded. Detailed description of the two tested mutation operators can be found in Section 3. Section 4 includes description of testing environments and applied measures while Section 5— obtained results and conclusions. A short discussion about the measure applied for the evaluation of the results is presented in Section 6. Final remarks are gathered in Section 7.

2 Artificial Immune Iterated Algorithm (AIIA)

The detailed description of AIIA can be found in [11]. The pseudo-code of the main loop of AIIA is given in Figure 1 below. The symbol \mathbf{x}_i represents i -th antibody, $\mathbf{x}_{i,k}$ — k -th mutated clone of the i -th antibody, $f(\mathbf{x}_i)$ — fitness of the i -th antibody to the antigen, and $\mathbf{x}_{i,c}^*$ is the best mutated clone of the i -th antibody, i.e. $\mathbf{x}_{i,c}^* = \arg \max_{\mathbf{x}_{i,k}, \forall k \in \{1, \dots, c\}} f(\mathbf{x}_{i,k})$ where c is the number of clones.

1. *Fitness evaluation.* For each antibody \mathbf{x}_i in the population P compute its fitness i.e. the value of the objective function $f(\mathbf{x}_i)$.
2. *Clonal selection.* Choose n antibodies with highest fitness to the antigen.
3. *Somatic hypermutation.*
 Make mutated clones $\mathbf{x}_{i,k}$, $k \in \{1, \dots, c\}$ for each antibody \mathbf{x}_i ,
 $i \in \{1, \dots, n\}$.
 The mutated clone $\mathbf{x}_{i,c}^*$ with highest fitness replaces the original antibody \mathbf{x}_i if $f(\mathbf{x}_{i,c}^*) > f(\mathbf{x}_i)$.
4. *Apoptosis.* Replace d weakest antibodies by randomly generated solutions.

Fig. 1. Pseudo-code of the main loop of AIIA

AIIA has five control parameters: $|P|$ — population size, n — size of the subpopulation activated for *clonal selection* procedure, c — number of mutated clones of the antibody (in general the number of clones for each of activated antibodies could be different. However in experiments presented below we simplified this rule and the number of clones of each antibody was the same), d — size of the subpopulation that undergo *apoptosis* procedure, and r_m — mutation range.

3 Mutations Based on Symmetric α -Stable Distributions

Authors of [7] point Gutowski's publication in 2001 [4] as the first paper treating the usefulness of α -stable distribution in heuristic global optimization algorithms. Since then there appeared more publications where properties of α -stable distributions as well as the results of application of α -mutation operators in evolutionary algorithms were widely discussed [5,7].

The α -stable distribution is controlled by four parameters: stability index α ($\alpha \in \langle 0 < \alpha \leq 2 \rangle$), skewness parameter β , scale parameter σ and location parameter μ . In symmetric version of this distribution (called $S\alpha S$, i.e. symmetric α -stable distribution) β is set to 0. For $\alpha = 2$ the $S\alpha S(\mu, \sigma)$ distribution reduces to the Gaussian $N(\mu, \sigma)$ and in the case of $\alpha = 1$ the Cauchy $C(\mu, \sigma)$ is obtained.

3.1 $S\alpha S(\mu, \sigma)$ Generator

In both tested mutation operators a $S\alpha S$ random numbers generator is applied. In [7] a theorem is presented where a formula for the simulation of standard α -stable symmetric random variable X for $\sigma = 1$ and $\mu = 0$ is given:

$$X = \begin{cases} \frac{\sin(\alpha V)}{(\cos(V))^{\frac{1}{\alpha}}} \left[\frac{\cos((\alpha-1)v)}{W} \right]^{\frac{1-\alpha}{\alpha}} & \text{if } \alpha \neq 1, \\ \tan(V), & \text{if } \alpha = 1. \end{cases} \tag{1}$$

V and W are independent random variables where: $V \sim U(-\frac{\pi}{2}, \frac{\pi}{2})$, and W is exponentially distributed with the mean 1. Thus a random variable $Z \sim S\alpha S(\mu, \sigma)$ is obtained by:

$$Z = \mu + \sigma X. \tag{2}$$

In our research the generator based on this formula (called $S\alpha S(\mu, \sigma)$ generator) was implemented and applied in the experiments¹.

3.2 Simple $S\alpha S(\mu, \sigma)$ Mutation Operator

The first of the tested mutation operators was a simple mutation where a new value of an l -th coordinate of a mutated clone $\mathbf{x}_{i,k}$ is calculated as follows:

$$\mathbf{x}_{i,k}[l] = \mathbf{x}_i[l] + S\alpha S(0, r_m) \cdot (dom_width_l/2).$$

where $\mathbf{x}_{i,k}[l]$ — the value of the l -th coordinate of the mutated clone $\mathbf{x}_{i,k}$, $\mathbf{x}_i[l]$ — the value of the l -th coordinate of the clone’s predecessor, r_m — the mutation range ($0 < r_m \leq 1$), and dom_width_l — a constant value which is equal to the distance between the upper and the lower boundary of the l -th dimension of the search domain.

If the value of $\mathbf{x}_{i,k}[l]$ is out of the domain then the remainder of $\mathbf{x}_{i,k}[l]$ and dom_width_l is evaluated as advised in [8]: $\mathbf{x}_{i,k}[l] = \mathbf{x}_{i,k}[l] \bmod dom_width_l$.

3.3 Modified opt-IA Mutation Operator

The second tested mutation operator originates from [3] where it was a component of the aiNet optimization algorithm. However in the version presented below the $S\alpha S(\mu, \sigma)$ generator was applied instead of the Gaussian random number generator. Thus the mutated value of an l -th coordinate of a clone $\mathbf{x}_{i,k}$ is calculated as follows:

$$\begin{aligned} \mathbf{x}_{i,k}[l] &= \mathbf{x}_i[l] + S\alpha S(0, \sigma_{i,l}), \\ \sigma_{i,l} &= r_m \cdot (dom_width_l/2) \cdot \exp(-f'(\mathbf{x}_i)). \end{aligned}$$

where $f'(\mathbf{x}_i)$ is the fitness of the clone’s predecessor normalized in $[0,1]$:

$$\begin{aligned} f'(\mathbf{x}_i) &= \frac{f(\mathbf{x}_i) - f_{min}}{(f_{max} - f_{min})}, \\ f_{max} &= \max_{\mathbf{x}_j, \forall j \in \{1, \dots, |P|\}} f(\mathbf{x}_j) \text{ and } f_{min} = \min_{\mathbf{x}_j, \forall j \in \{1, \dots, |P|\}} f(\mathbf{x}_j). \end{aligned}$$

¹ The C++ source code of the $S\alpha S(\mu, \sigma)$ generator (a MS Visual Studio .NET 2003 project) employed in the presented research is available at: <http://www.ipipan.waw.pl/~trojanow/alpha/>

4 Plan of Experiments and Applied Measures

Behavior of the algorithms was tested with six environments generated with two test-benchmarks. The first test-benchmark is a Test Case Generator [\[2\]](#) (or TCG) proposed in [\[10\]](#). We created four testing environments with TCG; two of them with cyclic changes and two with non-cyclic ones. In case of cyclic changes a full run of a single experiment includes 5 cycles of changes in the environment. In case of non-cyclic environments the total number of changes for the full run was 25. The second test-benchmark is a Moving Peaks Benchmark (or MPB) generator [\[2,6\]](#). Its description, sample parameters settings and a source code are available at the web page [\[1\]](#). We created two testing environments with MPB called scenario 1 and 2 [\[1\]](#). The total number of changes for the full run for each of the scenarios was also set to 25.

Table 1. Parameters of environments for six groups of experiments: TCG_{10c} and TCG_{20c}, TCG_{12nc}, TCG_{20nc}, MPB₅ and MPB₅₀

| <i>Environment</i> | TCG _{10c} | TCG _{20c} | TCG _{12nc} | TCG _{20nc} | MPB ₅ | MPB ₅₀ |
|------------------------------|--------------------|--------------------|---------------------|---------------------|------------------|-------------------|
| <i>No. of dimensions</i> | 2 | 2 | 2 | 2 | 5 | 5 |
| <i>Environment type</i> | 10×10 | 10×10 | 6×6 | 10×10 | 1 | 2 |
| <i>No. of varying optima</i> | 10 | 20 | 12 | 20 | 5 | 50 |
| <i>No. of iterations</i> | 1000 | 2000 | 500 | 500 | 500 | 500 |

Six groups of experiments were performed. Table [1](#) shows the settings of each of the groups. The first row called *No. of dimensions* shows the number of dimensions of the search space where the optimization is performed. *Environment type* shows the identifier of the testing environment. In case of TCG the identifier describes the landscape for the 2-dimensional search space, e.g. 10×10 means that we have a "chessboard" with 100 fields (10 by 10) each with a single hill in the center. In case of MPB the identifier is a number of scenario. *No. of varying optima* shows the number of varying hills, peaks or cones in the optimization landscape. *No. of iterations* shows the number of iterations performed in one full run of a single experiment. In case of cyclic changes this number is equal to: number of cycles of changes multiplied by the number of changes in a cycle (i.e. no. of varying optima) and by the number of iterations between changes (e.g. for TCG_{10c} it is: 5×10×20).

To evaluate the results, we used a measure called offline error which represents the average deviation of the best individual from the optimum evaluated since the last change of the fitness landscape. Every time the solution's fitness is evaluated, an auxiliary variable is increased by the value which is the deviation of the best

² Figures of sample environments generated with TCG are available at: <http://www.ipipan.waw.pl/~stw/ais/environment/env.html>

solution evaluated since the last change including the one just evaluated as well. When the experiment is finished the sum in the variable is divided by the total number of evaluations and returned as the offline error. Every experiment was repeated 100 times and the mean was calculated.

5 Results of Experiments

For each of the testing environments a set of tests was performed where the influence of three selected parameters of the algorithm on the offline error was observed. The first parameter is α in the $S\alpha S(\mu, \sigma)$ generator applied in the mutation operators. Tests with the following values of the α parameter were performed: $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.75, 1, 1.5, 1.75, 1.9, 1.95, 2\}$. The second parameter, r_m varied from 0.01 to 0.99 with step 0.02. The third parameter, the number of activated antibodies varied from 2 to 20 with step 1. To satisfy the assumption of the same or almost the same number of fitness evaluations between changes in the fitness landscape from one side and the assumption of constant number of iterations between changes from the other side the number of clones had to depend on size of the set of activated antibodies. For a set of size 2 there were 99 clones, for 3 – 66, 4 – 50, 5 – 40, 6 – 33, 7 – 29, 8 – 25, 9 – 22, 10 – 20, 11 – 18, 12 – 17, 13 – 16, 14 – 15, 15 – 14, 16 – 13, 17 – 12, 18 – 11, 19 – 11, 20 – 10.

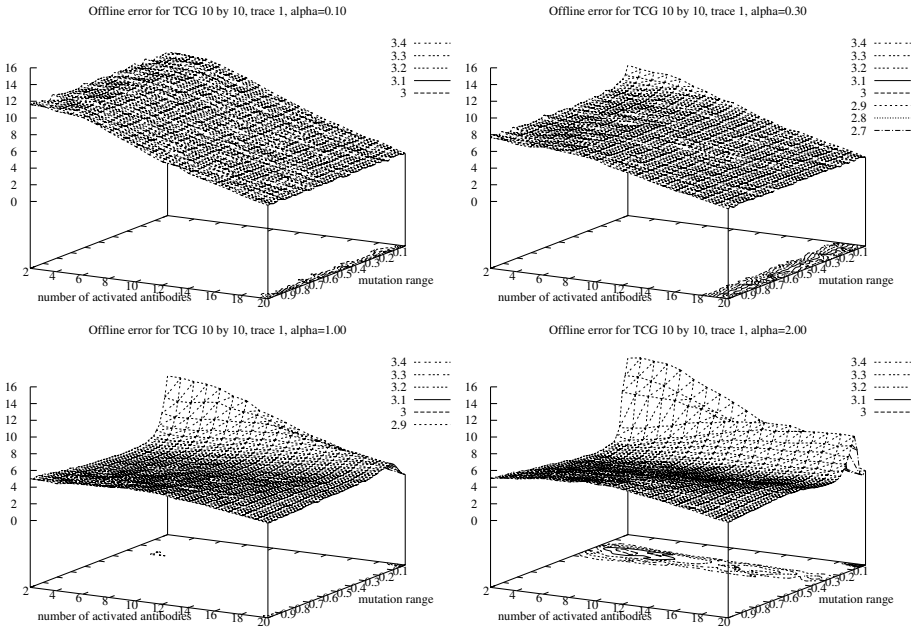


Fig. 2. Offline error obtained by AIIA for the tested parameters: r_m (mutation range) and number of activated antibodies. Four graphs for the values of α : 0.1, 0.3, 1 and 2.

Sample results of the algorithm with the modified opt-IA mutation for four selected values of α and for the selected test-case TCG_{10c} are presented in Figure 2. Similar offline error landscapes were obtained for all the testing configurations. For each of the tested values of α the best values of offline error were found. They are presented in Tables 2 and 3.

Table 2. Best values of offline error obtained by AIIA with simple $S\alpha S$ mutation

| α : | TCG _{10c} | TCG _{20c} | TCG _{12nc} | TCG _{20nc} | MPB ₅ | MPB ₅₀ |
|---------------|--------------------|--------------------|---------------------|---------------------|------------------|-------------------|
| 0.10 | 9.24 | 21.18 | 11.22 | 18.55 | 50.28 | 24.27 |
| 0.20 | 9.24 | 21.18 | 11.19 | 18.54 | 50.28 | 24.27 |
| 0.30 | 9.24 | 21.18 | 11.18 | 18.57 | 50.28 | 24.27 |
| 0.40 | 9.24 | 21.18 | 11.26 | 18.49 | 50.28 | 24.27 |
| 0.50 | 9.32 | 21.14 | 11.17 | 18.57 | 50.28 | 24.27 |
| 0.75 | 9.32 | 21.14 | 11.31 | 18.66 | 50.28 | 24.11 |
| 1.00 (Cauchy) | 9.26 | 21.16 | 11.26 | 18.51 | 50.74 | 24.36 |
| 1.50 | 9.32 | 21.14 | 11.30 | 18.71 | 50.28 | 24.27 |
| 1.75 | 9.24 | 21.18 | 11.20 | 18.70 | 50.28 | 24.27 |
| 1.90 | 9.24 | 21.18 | 11.27 | 18.64 | 50.28 | 24.27 |
| 1.95 | 9.24 | 21.18 | 11.24 | 18.55 | 50.28 | 24.27 |
| 2.00 (Gauss) | 9.30 | 21.07 | 11.22 | 18.64 | 50.86 | 24.66 |

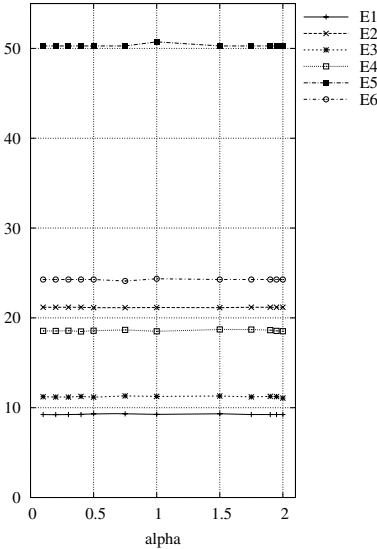
Results of experiments for the first type of mutation, i.e. simple $S\alpha S(\mu, \sigma)$ mutation are presented in Table 2 and their graphical representation can be found in Figure 3.a. It can be seen that there is no relevant relationship between the value of α of the $S\alpha S(\mu, \sigma)$ generator and the algorithm's efficiency. For all the tested values of α the obtained results are quite similar to each other. In comparison to the results presented in other publications about non-stationary optimization they are not very impressive.

The second group of results for the modified opt-IA mutation is presented in Table 3 and in Figure 3.b. These results are much better and show more regular relationship than the results for the first type of mutation. Starting with $\alpha = 0.1$ it can be seen that the error decreases with the growth of the α value. For α between 0.3 and 1.0 the graphs are getting stable. For two test-cases an improvement of the results can be observed till the end of the tested interval of α values while for four remaining test-cases there exist optimum value of α . The optimum α values for the four tasks are respectively: TCG_{10c} — 0.3, TCG_{12nc} — 1.0, MPB₅ — 0.5, and MPB₅₀ — 0.5. It completely confirms observations known from the literature, where in many cases the Cauchy mutation was indicated as the more efficient than the Gaussian one.

Table 3. Best values of offline error obtained by AIIA with modified opt-IA mutation

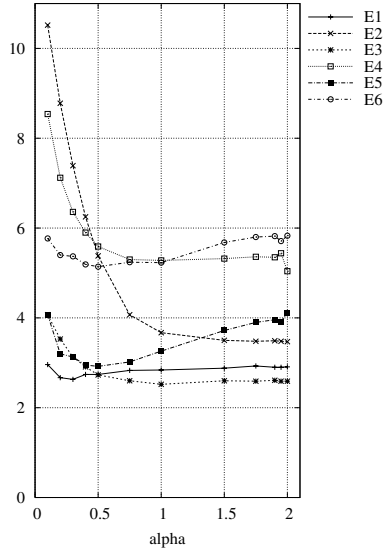
| α : | TCG _{10c} | TCG _{20c} | TCG _{12nc} | TCG _{20nc} | MPB ₅ | MPB ₅₀ |
|---------------|--------------------|--------------------|---------------------|---------------------|------------------|-------------------|
| 0.10 | 2.96 | 10.52 | 4.07 | 8.54 | 4.07 | 5.77 |
| 0.20 | 2.67 | 8.78 | 3.53 | 7.12 | 3.20 | 5.40 |
| 0.30 | 2.63 | 7.39 | 3.13 | 6.36 | 3.13 | 5.37 |
| 0.40 | 2.74 | 6.25 | 2.91 | 5.90 | 2.95 | 5.19 |
| 0.50 | 2.74 | 5.38 | 2.73 | 5.59 | 2.92 | 5.14 |
| 0.75 | 2.83 | 4.07 | 2.60 | 5.30 | 3.02 | 5.24 |
| 1.00 (Cauchy) | 2.84 | 3.67 | 2.52 | 5.28 | 3.26 | 5.23 |
| 1.50 | 2.88 | 3.50 | 2.60 | 5.32 | 3.72 | 5.68 |
| 1.75 | 2.93 | 3.48 | 2.59 | 5.36 | 3.90 | 5.80 |
| 1.90 | 2.90 | 3.49 | 2.61 | 5.35 | 3.96 | 5.82 |
| 1.95 | 2.90 | 3.48 | 2.59 | 5.44 | 3.92 | 5.71 |
| 2.00 (Gauss) | 2.91 | 3.47 | 2.59 | 5.04 | 4.11 | 5.83 |

AIIA with alpha-stable simple mutation - offline error



a)

AIIA with alpha-stable optIA mutation - offline error



b)

Fig. 3. Comparison of the offline error obtained with two types of mutation. E1 represents best i.e. least values of error for TCG_{10c}, E2 – for TCG_{20c}, E3 – for TCG_{12nc}, E4 – for TCG_{20nc}, E5 – for MPB₅, and E6 – for MPB₅₀.

T-student's test was used for analysis of the significance of differences between average values of offline error in the neighboring rows of Table 3. Calculated t -values are presented in Table 4. P value less than 0.05 was considered significant. For 198 degrees of freedom and the assumed level of significance the tabulated critical value t_α equals 1.9720.

Table 4. An analysis of the significance of differences between average values of offline error obtained by AIIA with modified opt-IA mutation: calculated t -values. t -values greater than critical value t_α are underlined.

| compared α : | TCG _{10c} | TCG _{20c} | TCG _{12nc} | TCG _{20nc} | MPB ₅ | MPB ₅₀ |
|---------------------|--------------------|--------------------|---------------------|---------------------|------------------|-------------------|
| 0.10-0.20 | <u>2.8891</u> | <u>14.5448</u> | <u>5.1021</u> | <u>5.0997</u> | <u>8.6804</u> | 1.5778 |
| 0.20-0.30 | 0.3503 | <u>12.3526</u> | <u>5.3890</u> | <u>4.7219</u> | 0.5942 | 0.3347 |
| 0.30-0.40 | 0.7108 | <u>11.5419</u> | <u>2.1186</u> | <u>1.3052</u> | 0.1475 | 0.7035 |
| 0.40-0.50 | 0.2596 | <u>11.0943</u> | <u>2.3398</u> | <u>2.0001</u> | 0.1345 | 0.0880 |
| 0.50-0.75 | 1.1799 | <u>17.6733</u> | <u>2.4652</u> | 1.9183 | 0.9053 | 0.1719 |
| 0.75-1.00 | 1.4025 | <u>6.2833</u> | 0.5804 | 0.5027 | <u>2.3375</u> | 0.9176 |
| 1.00-1.50 | 1.8801 | 1.8569 | <u>2.2835</u> | 0.9993 | <u>3.0922</u> | 0.2053 |
| 1.50-1.75 | 0.9431 | 0.8323 | 0.9960 | 1.4427 | 0.0000 | 0.6077 |
| 1.75-1.90 | 0.6418 | 1.4487 | 0.8270 | 0.4763 | 1.1506 | 0.0000 |
| 1.90-1.95 | 0.1551 | 0.9740 | 0.1229 | 0.2277 | 1.3545 | 0.1338 |
| 1.95-2.00 | 0.1471 | 0.1560 | 1.4745 | <u>2.3474</u> | 0.7934 | 0.6347 |

In Table 4 it can be seen that in many cases calculated t -values do not exceed assumed critical t_α -value so some of the pairs of means can not be considered as significantly different. For most cases the last significant difference between the means of offline error is in the middle of the tested interval of the α parameters. In all the cases except TCG_{20nc} the differences between pairs obtained for α close to 2 were not statistically significant. For MPB₅₀ none of the pairs of means differs significantly and for TCG_{10c} there was just one such a pair. From the other side the differences between pairs obtained for α close to 0 were usually significant.

6 Indeterminism in the Offline Error Measure

When we take a closer look at the formula of evaluation of the offline error it can be noticed that there is a kind of indeterminism in the way of its application and thus a set of different values of the error can be obtained for the same set of solutions. Depending on the order of evaluation of the solutions the offline error can be lower if we start from the best of them or higher if we start from

the worst ones and the best are evaluated last. Difference in the obtained value of the error depends on the difference between the fitness of the best individual in the population and the fitness of the worst one and of the distribution of the better and worse fitness values over the population.

In all our experiments presented above the set of solutions was never sorted for the better tuning of the results. However we wanted to check the difference in the obtained results for the case where after every change in the fitness landscape the set of solutions was sorted from the best to the worst one before the reevaluation of them. Additionally we also wanted to check stability of the results obtained from the series of 100 repetitions of the tests. For a single configuration of the algorithm and for the selected test-case MPB₅₀ we performed a hundred tests where each test consisted of a hundred experiments. The result of a single test is a single value which is a mean of all the offline errors in the series. For the results of the hundred tests the following values were evaluated: mean, median, minimum value, maximum value and standard deviation (Table 5).

Table 5. Comparison of offline error obtained by AIIA with modified opt-IA mutation ($\alpha = 0.5$) without and with sorting of population for a series of 100 tests

| type: | mean | median | min. val. | max. val. | std. dev. |
|----------------------|--------|--------|-----------|-----------|-----------|
| <i>without</i> sort. | 5.5086 | 5.505 | 5.10 | 5.88 | 0.16960 |
| <i>with</i> sort. | 5.5015 | 5.500 | 5.09 | 5.87 | 0.16979 |

Results in Table 5 show that there is a difference in benefit to the version *with* sorting however it is a very low difference. In case of mean values it is on the third digit after the decimal point. For both versions the values of mean and median are similar to each other which means that the distribution is symmetric. Additionally the estimation of a single experiment's standard deviation can be obtained from last column of the table. Since the test is an average of 100 experiments, the standard deviation of a single experiment is equal to $\sqrt{100} = 10$ times greater.

Finally let us note that the respective value of offline error in Table 3 is higher than the minimal value of offline error in Table 5 and lower than the mean value. Thus the result in Table 3 can be classified as a rather good one. Unfortunately we do not have a similar knowledge about all the remaining values in Tables 2 and 3. In spite of this the significance of tendencies observed in these tables and in Figure 3 is not depreciated.

7 Conclusions

In this paper applicability of a version of the clonal selection algorithm called AIIA [11] was studied. We tested the algorithm equipped with two types of

mutation, both using the $S_{\alpha S}(\mu, \sigma)$ generator. Obtained results showed that the second of them, the modified opt-IA mutation is much better than the first one, i.e. simple mutation.

The results also showed clearly that the parameter α of the random number generator can be used to advantage as yet another parameter for control of the clonal selection based algorithm. This parameter can significantly influence the algorithm's behavior and efficiency.

References

1. J. Branke. The moving peaks benchmark. URL: <http://www.aifb.uni-karlsruhe.de/~jbr/MovPeaks/movpeaks/>.
2. J. Branke. Memory enhanced evolutionary algorithm for changing optimization problems. In *Proc. of the Congress on Evolutionary Computation*, volume 3, pages 1875–1882. IEEE Press, Piscataway, NJ, 1999.
3. L.N. de Castro and J. Timmis. An artificial immune network for multimodal function optimization. In *Proc. of the Congress on Evolutionary Computation*, volume 1, pages 699–704. IEEE Press, Piscataway, NJ, 2002.
4. M. Gutowski. Levy flights as an underlying mechanism for a global optimization algorithm. In *Proc. 5th National Conf. on Evolutionary Computation and Global Optimisation*, pages 79–86. Warsaw Univ. of Technology Publishing House, 2001.
5. C.-Y. Lee and X. Yao. Evolutionary programming using mutations based on the levy probability distribution. *IEEE Trans. on Evolutionary Computation*, 8(1):1–13, 2004.
6. R. W. Morrison and K. A. De Jong. A test problem generator for non-stationary environments. In *Proc. of the Congress on Evolutionary Computation*, volume 3, pages 1859–1866. IEEE Press, Piscataway, NJ, 1999.
7. A. Obuchowicz and P. Pretki. Phenotypic evolution with a mutation based on symmetric α -stable distributions. *Int. J. Appl. Math. Comput. Sci.*, 14(3):289–316, 2004.
8. M. Purchla, M. Malanowski, P. Terlecki, and J. Arabas. Experimental comparison of repair methods for box constraints. In *Proc. 7th National Conf. on Evolutionary Computation and Global Optimisation*, pages 135–142. Warsaw Univ. of Technology Publishing House, 2004.
9. K. Trojanowski. Clonal selection principle based approach to non-stationary optimization tasks. In *Proc. 9th National Conf. on Evolutionary Computation and Global Optimisation*, number 156 in Prace Naukowe, Elektronika, pages 375–384. Warsaw Univ. of Technology Publishing House, 2006.
10. K. Trojanowski and Z. Michalewicz. Searching for optima in non-stationary environments. In *Proc. of the Congress on Evolutionary Computation*, volume 3, pages 1843–1850. IEEE Press, Piscataway, NJ, 1999.
11. K. Trojanowski and S. T. Wierzchoń. Studying properties of multipopulation heuristic approach to non-stationary optimisation tasks. In *IIS 2003: Intelligent Information Processing and Web Mining*, Advances in Soft Computing, pages 23–32. Springer Verlag, 2003.
12. S.T. Wierzchoń. Function optimization by the immune metaphor. *Task Quarterly*, 6(3):493–508, 2002.

Minimizing Cycle Time of the Flow Line – Genetic Approach with Gene Expression

Paweł Dąbrowski, Jarosław Pempera, and Czesław Smutnicki

Wrocław University of Technology, Institute of Computer Engineering, Control and Robotics, Janiszewskiego 11-17, 50-372 Wrocław, Poland
{jaroslaw.pempera, czeslaw.smutnicki}@pwr.wroc.pl

Abstract. This paper deals with the flow-shop scheduling problem with no-store policy and minimal cycle time criterion. The model and some properties of the problem have been presented. To solve the problem, we propose new genetic algorithm equipped with auxiliary gene expression mechanism, which creates offspring using genetic information from both parents as well as asleep information from ancestors (grand-father, grand grandfather). The presented computational tests proved superiority of the proposed approach over traditional, basic GA scheme.

1 Introduction

Cyclic manufacturing is the fundamental strategy of systems providing mixture of various products, in long production series, with slowly changeable production profile. Cyclic system provides on the output, every time period called *cycle time*, the fixed mixture of products, which composition depends on medium- and long-time orders of clients. By minimizing cycle time we are able to maximize the productivity of the system as well as the degree of machines utilization. The system efficiency can be additionally improved by limitation or elimination storage of semi-products between production stages. This leads to constraints "no store" and "limited store" quite often required in cyclic manufacturing.

Cyclic manufacturing, modelled in terms of deterministic scheduling problems with storage constraints, belongs to the class of the hard discrete optimization cases. Not only quite non-trivial numeric procedures, that find the minimal cycle time and the corresponding schedule, but also the lack of problem-specific properties makes the search process particularly hard. That's why studies carried out up till now did not go beyond basic production structures (as an example flow line, which refers to flow-shop scheduling problem), whereas nobody defines and analyses cycle time for more complex system, as an example job-shop.

Strong NP-hardness of these problems limits the applicability of exact optimization methods to instances of small size only. Therefore, the intelligent fast approximate methods become foreground tasks in the scheduling research. This type methods base chiefly on two-level decomposition of the problem: (A) find the optimal sequence of jobs in a cycle (upper level) and (B) find the minimal cycle time for fixed sequence of jobs (lower level). Generally (B) can be solved

by using a linear programming method, a min flow problem or a problem of finding specific paths in a graph. Unfortunately, the lack of special properties for (B), eliminates most of local search approaches (as excessively time consuming) and pointed out genetic approach (GA) and simulated annealing (SA) as the potentially promising, see e.g. [1]. We use the former one.

Advantages and disadvantages of GA [2] are commonly known. In order to obtain the best effect, GA needs to be tuned in time consuming tests. Moreover, *binary coding* of chromosome finally appears unusable for scheduling problems. This have implied certain non-homogeneity in chromosome coding techniques, that finally tends towards the concept: *solution = combinatorial object = individual = genotype* (assumption *individual = genotype* simply reduces the number of tuning parameters). This, in order, increases the role of problem-specific genetic operators being the "source of progress". Thus, the artificial evolution creates distracted individuals, in accordance with pure democratic and multi-directed policy of the Nature. Such democracy is superfluous in scheduling problems where search history can efficiently influence on further search directions. The Nature owns similar mechanism since, in fact, the individual is determined by the collection of phenotype features, which follows from the huge number of genes, carried out from a generation to the next generation, frequently asleep, expressing indirectly data from the history of generation development. This leads to the idea: *solution = individual = phenotype \neq genotype* presented in this paper.

2 The Problem

In this section we introduce the cyclic scheduling problem as the problem of combinatorial optimization. The set of tasks $J = \{1, 2, \dots, N\}$ has to be processed on machines, indexed $1, 2, \dots, m$, organized in a line structure. Single task reflects one final product manufactured. Every task is performed in m subsequent stages; task routes are identical for all tasks. Stage i is performed by machine i .

Next, there is no storage place neither in production stockroom nor in buffers between subsequent machines. It means, that task completed on machine i can be moved to machine $i + 1$ if only $i + 1$ is free, otherwise this task blocks machine i until machine $i + 1$ will be released. Then, every task $j \in J$ may be perceived as the sequence of m operations $O_{1j}, O_{2j}, \dots, O_{mj}$ performed on machines in turn. Operation O_{ij} reflects processing of task j on machine i with processing time $p_{ij} > 0$. Once started, operation cannot be interrupted. Each machine can execute at most one task at a time, each task can be processed on at most one machine at a time. Systems restricted by limited buffers capacity or buffers absence are subjects of interest of many researchers [3,4,5,6].

Let us assume that set J consists of t different types of tasks, with cardinality of every type denoted by r_1, \dots, r_t , respectively; task of identical type have identical processing times of individual operations. Thus, we have $N = \sum_{i=1}^t r_i$. If there exists integer $c > 1$, being the greatest common division of integers r_1, \dots, r_t , then set J may be split into c identical subsets, every of them containing $n = r_1/c + \dots + r_t/c$ tasks. Each of this subset is called the *minimal part*

set (MPS). After completing k -th MPS (i.e. its task in a fixed order), system performs $(k + 1)$ -st MPS (tasks processed in the same order), $k = 1, 2, \dots$

We will focus only on tasks from single MPS. It is shown that feasible processing order on every machine must be identical, [4]. Thus, order of tasks from single MPS may be defined as permutation π of elements from set $\{1, \dots, n\}$. The processing order of all tasks from J can be obtained by c -times concatenation of π . Schedule of tasks from k -th MPS can be described by matrix $[S^k]_{m \times n}$ where $S_{i,j}^k$ is the time moment when processing of task j starts on machine i . Feasible schedule $[S^k]_{m \times n}$, for given π , must satisfy the following set of constraints

$$S_{i,\pi(j)}^k + p_{i,\pi(j)} \leq S_{i+1,\pi(j)}^k, \quad i = 1, \dots, m-1, j = 1, \dots, n, \quad (1)$$

$$S_{i,\pi(j)}^k + p_{i,\pi(j)} \leq S_{i,\pi(j+1)}^k, \quad i = 1, \dots, m, j = 1, \dots, n-1, \quad (2)$$

$$S_{i+1,\pi(j)}^k \leq S_{i,\pi(j+1)}^k, \quad i = 1, \dots, m-1, j = 1, \dots, n-1, \quad (3)$$

$$S_{i,\pi(n)}^k + p_{i,\pi(n)} \leq S_{i,\pi(1)}^{k+1}, \quad i = 1, \dots, m, \quad (4)$$

$$S_{i+1,\pi(n)}^k \leq S_{i,\pi(1)}^{k+1}, \quad i = 1, \dots, m-1, \quad (5)$$

for $k = 1, \dots$. Constraint (1) comes from technological processing order, while (2) from unitary machine throughput. Constraint (3) models store limitations - processing of task $\pi(j+1)$ on the machine i may start not earlier than processing of task $\pi(j)$ starts on the machine $i+1$; i. e. machine i must be released by task $\pi(j)$. Constraints (4) and (5) model time relations between processing of last task from k -th MPS and the first task from $k+1$ -st MPS. These constraints are fully analogous to (2) and (3). Assumption about cyclic production requires, that there exist constant T , called the *cycle time*, such that

$$S_{i,\pi(j)}^{k+1} = S_{i,\pi(j)}^k + T, \quad i = 1, \dots, m, j = 1, \dots, n, k = 1, 2, \dots \quad (6)$$

Obviously, T depends on π . Actually, we are interested in minimal possible value T (set individually for fixed schedule π), such that constraints (1)-(6) are fulfilled. This value will be called *minimal cycle time* and denoted by $T(\pi)$.

The optimization problem is as follows. Let Π denote the set of all permutations on the set $\{1, \dots, n\}$. We want to find permutation $\pi^* \in \Pi$, such that

$$T(\pi^*) = \min_{\pi \in \Pi} T(\pi). \quad (7)$$

3 Minimal Cycle Time

This section aim is to provide methods of finding $T(\pi)$ for fixed π . Let π is fixed and T is a cycle time defined by (6). Substituting (6) to (1)-(5) we get set of constraints defined for single MPS, identical for all $k = 1, 2, \dots$. This means that index k may be omitted in $S_{i,j}^k$. Thus, $T(\pi)$ and schedule $S_{i,j}$, $i = 1, \dots, m, j = 1, \dots, n$ can be obtained by solving the following linear programming problem

$$T(\pi) = \min_{T, S_{i,j}} T \quad (8)$$

$$S_{i,\pi(j)} + p_{i,\pi(j)} \leq S_{i+1,\pi(j)}, \quad i = 1, \dots, m-1, \quad j = 1, \dots, n, \quad (9)$$

$$S_{i,\pi(j)} + p_{i,\pi(j)} \leq S_{i,\pi(j+1)}, \quad i = 1, \dots, m, \quad j = 1, \dots, n-1, \quad (10)$$

$$S_{i+1,\pi(j)} \leq S_{i,\pi(j+1)}, \quad i = 1, \dots, m-1, \quad j = 1, \dots, n-1, \quad (11)$$

$$S_{i,\pi(n)} + p_{i,\pi(n)} \leq S_{i,\pi(1)} + T, \quad i = 1, \dots, m, \quad (12)$$

$$S_{i+1,\pi(n)} \leq S_{i,\pi(1)} + T, \quad i = 1, \dots, m-1, \quad (13)$$

too time consuming to be applied in any iterative approximate algorithm. Therefore, we are looking for other less expensive methods.

An alternative methodology was presented in [7], where schedule S_{ij} , $i = 1, \dots, m$, $j = 1, \dots, n$ and $T(\pi)$ was obtained by solving a problem of minimal flow in the specific net. Considering the size of the net $O(nm)$ and computational complexity of the flow algorithm, we obtain still too expensive method having the computational complexity $O(n^3m^3)$.

We propose, in the sequel, the efficient method, with computational complexity $O(nm^2)$, based on paths in a graph. From (12)-(13) it follows, that

$$T(\pi) \geq LB_T(\pi) = \max_{1 \leq i \leq m} [\max\{S_{i,\pi(n)} + p_{i,\pi(n)}, S_{i+1,\pi(n)}\} - S_{i,\pi(1)}] \quad (14)$$

where $S_{m+1,j} = S_{m,j}$, $j = 1, \dots, n$.

In order to find $LB_T(\pi)$ we build rectangular grid graph with $m+1$ rows and $n+1$ columns, similarly as in [6]. It has the form of $G(\pi) = (V, R \cup F(\pi))$ with set of nodes V and set of arcs $R \cup F(\pi)$, where $V = \{1, \dots, m+1\} \times \{1, \dots, n+1\}$, $R = \bigcup_{k=1}^m \bigcup_{j=1}^{n+1} \{(k, j), (k+1, j)\}$, $F(\pi) = \bigcup_{k=1}^m \bigcup_{j=1}^n \{(k+1, \pi(j)), (k, \pi(j+1))\}$ respectively. Node $(i, j) \in V$ represents operation i of task $\pi(j)$. Node weight $(i, j) \in V$ is $p_{i,\pi(j)}$. Arcs from set R model constraints (10) and (12), and have weight equal 0. Arcs from set $F(\pi)$ model constraints (11) and (13); every arc has weight minus $p_{k+1,\pi(j)}$. Column j of this graph corresponds to tasks $\pi(j)$, $j = 1, \dots, n$. Row k corresponds to machine k , $k = 1, \dots, m$. Row $(m+1)$ -st represents virtual machine $m+1$, with $p_{m+1,j} = 0$, $j = 1, 2, \dots, n$. Column $n+1$ represents virtual task being copy of the first task, namely $\pi(n+1) = \pi(1)$.

Let D_i , $i = 1, \dots, m$, is the length of the longest path from node $(i, 1)$ to node $(i, n+1)$, including $p_{i,\pi(1)}$ but excluding $p_{i,\pi(n+1)}$, in $G(\pi)$. We proved that

Property 1. $LB_T(\pi) = D_{max} = \max_{1 \leq i \leq m} D_i$.

Property 2. There exists solution of (8)-(13), such that $T(\pi) = LB_T(\pi)$.

Notice, many local search algorithms need only $T(\pi)$, but S_{ij} , $i = 1, \dots, m$, $j = 1, \dots, n$. We have provided also an efficient method of finding the schedule S_{ij} , $i = 1, \dots, m$, $j = 1, \dots, n$, on the base of $T(\pi)$, however in the context of proposed next GA method it is of less importance.

4 Genetic Algorithm

Traditional scheme of genetic algorithms is a significant simplification of complex natural evolution process, where crossing is the base mechanism to produce

individuals with diverse features. In fact, crossing enables exchange of basic genetic information contained in DNA, which coded origin of rRNA, tRNA, mRNA and enzymes responsible for proteins biosynthesis producing material of life. For organism originated in this way, its external features are assessed according its adaptation to environment (estate). Phenotype feature (external) is determined usually by many genes interacting each other through so called gene expression mechanism. One can say, that information contained in DNA is in some meaning excessive (redundant) and provide information about history of given species. In the paper [8], essentially different model of phenotype representation was proposed for the Travelling Salesman Problem, joined with specific gene expression algorithm based on some heuristic method providing phenotype.

We assume that solution corresponds to phenotype, since it fitness is evaluated. For our problem the processing order of tasks in the cycle is represented by a permutation, so phenotype is a permutation on the set $\{1, \dots, n\}$. To hold genetic information coming from previous generations, we propose to code individual genotype as three permutations. The first one is agreeable with current individual phenotype, while the second and third are phenotypes of individuals father and grandfather, respectively. Obviously, in this way we can store genetic information coming from even older generations.

One of the most effective crossing operator for permutation chromosomes is the operator with partial matching PMX, [9]. For our needs, we propose new operator E adapted immediately from PMX to our extended representation of the chromosome. More precisely, in E pure crossing of individuals is done in following way: the first permutation of the descendant coming from parents by application of PMX, the second permutation is obtained by copying fathers phenotype, whereas the third one is got by copying the second permutation of the father, which in turn is identical with phenotype of grandfather (precisely father of the father). The second mirror descendant inherits by female line. The behaviour of E operator is presented in the example from Fig. 1. Let randomly selected cut points on the first permutation are $a = 2$, $b = 5$. For mother (first individual on the Fig. 1a) and father (second individual on the Fig. 1a), points a and b determines sequence of task transpositions $7 \leftrightarrow 5$, $1 \leftrightarrow 10$, $4 \leftrightarrow 3$, $2 \leftrightarrow 7$. The descendant is created by copying the first permutation of the father and exchanging elements listed in the above sequence of transpositions. The second and third permutations are inherited accordingly from father. Analogously, by changing roles of parents the second descendant can be generated.

Gene expression algorithm determines phenotype of an individual observable immediately after crossing, in our case on the base of three stored permutations. Then newly created phenotype is stored in the first permutation of this individual. The phenotype is created in two stages. In the first stage, there is modified position of elements, which comes from father and their positions have been fix during sequence of transpositions (underlined positions in the figure). Let us call these element as movable. Next, every pair of adjacent elements from fathers phenotype is taken into consideration. Let (x, y) is one of such pairs, then if x and y are movable elements, y is moved directly behind x in phenotype of

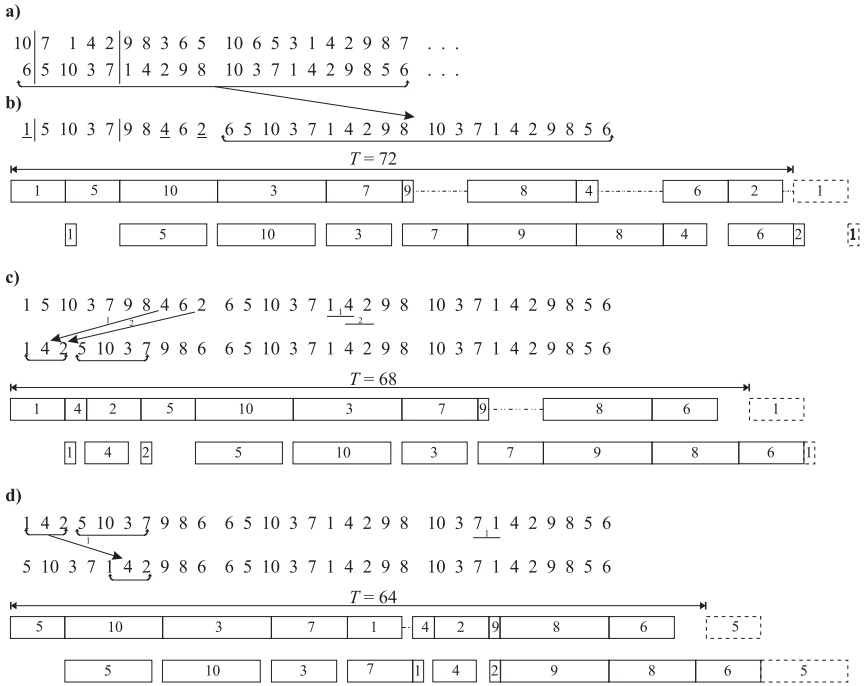


Fig. 1. Genetic engineering: (a) parents, (b) descendant, (c) gene expression stage one, (d) gene expression stage two. Corresponding Gantt charts and cycle times are included.

newly created individual. In our example, as result of such operation, element 4 is moved behind 1 and then element 2 is moved behind 4 (see Fig. 1c). At the begin of the second stage, descendant phenotype is split into chunks. Every chunk contains as much elements as possible which come from the same parent, and are performed in the same order in parents phenotype. In the second stage every adjacent pair from grandfather phenotype is analyzed. Let (x, y) be one of such pairs, then, if x is the last element in some segment and y is the first in other one, all elements from the second are put in behind element x respecting order they existed in this fragment. In Fig. 1d, as result of second stage, fragment $(1, 4, 2)$ is moved directly behind the last element of fragment $(5, 10, 3, 7)$.

5 Numerical Experiment

In order to examine the influence of the excessive coding with gene expression mechanism on the algorithm efficiency, algorithm from [10] has been implemented. It checks successive NGEN generations, works on the population POP of individuals, the size of the population is POPSIZE. The starting population is

generated at random. At a iteration, the fitness function $fit(t)$ of an individual $t \in POP$ is calculated using

$$fit(t) = \frac{\max_{j \in POP} [T(F(j)) - T(F(t))]}{\sum_{k \in POP} \max_{j \in POP} [T(F(j)) - T(F(k))]}, \quad t \in POP, \quad (15)$$

where $F(t)$ denotes the phenotype of individual t , and T is the cycle time. Then, for individual, the probability of its choosing is calculated as follows

$$pr(t) = \frac{fit(t)}{\sum_{k \in POP} fit(k)}, \quad t \in POP. \quad (16)$$

From population POP , pairs of parents are selected to reproduction according to roulette rule. Every pair is crossed with probability $PCROSS$ and or every individual is mutated with probability $PMUTE$. The mutation consists of exchange of two elements placed on two randomly selected positions.

New population contains newly created descendants and the best fitted individual from parent generation. In case of low diverse of the generation, i.e. if more than $POPFIT$ individuals match each other, population is replaced with the population generated randomly, in the same way as initial population. Fitness of individual $i \in POP$ means, that $\max_{j \in POP} (fit(j) - fit(i)) \leq FITTOLER$. Algorithm returns the best permutation generated during its working, i.e. permutation with minimal cycle time. Similarly like in [10], algorithm has been run with following parameters: $POPSIZE = 95$, $NGEN = 1000$, $FITTOLER = 1E - 10$, $POPFIT = 60$, $PCROSS = 0.25$, $PMUTE = 0.009$.

Algorithm, coded in C++, was tested using first six group of benchmarks of Taillard [11]. Genetic algorithm has been implemented with three crossover operators. The first implementation GA-PMX uses only PMX, the second one GA-EO uses PMX and the first stage of gene expression algorithm. The last algorithm GA-E uses crossover operator PMX and the full two-phase gene expression algorithm. Every variant of the algorithm was run 10 times for every test instance. For every problem instance, for every tested algorithm, (GA-PMX, GA-EO, GA-E) and for every from among 10 runs of the algorithm, the following

Table 1. Comparison of crossing operators efficiency

| Group $n \times m$ | GA-PMX | | | | GA-OE | | | | GA-E | | | |
|-----------------------|--------|------|------|------|-------|------|------|------|------|------|------|------|
| | AB | SB | MINB | MAXB | AB | SB | MINB | MAXB | AB | SB | MINB | MAXB |
| 20×5 | 2.95 | 0.29 | 1.29 | 4.41 | 2.01 | 0.26 | 0.56 | 3.27 | 1.12 | 0.25 | 0.01 | 2.57 |
| 20×10 | 2.48 | 0.33 | 0.82 | 4.13 | 2.18 | 0.26 | 0.95 | 3.56 | 1.49 | 0.26 | 0.13 | 2.83 |
| 20×20 | 1.81 | 0.20 | 0.79 | 2.79 | 1.43 | 0.18 | 0.48 | 2.42 | 1.13 | 0.21 | 0.07 | 2.26 |
| 50×5 | 2.97 | 0.24 | 1.68 | 4.21 | 2.15 | 0.28 | 0.67 | 3.52 | 1.52 | 0.26 | 0.13 | 2.89 |
| 50×10 | 2.46 | 0.23 | 1.26 | 3.83 | 1.65 | 0.25 | 0.43 | 3.03 | 1.35 | 0.24 | 0.16 | 2.59 |
| 50×20 | 1.50 | 0.14 | 0.72 | 2.28 | 1.16 | 0.17 | 0.31 | 2.03 | 1.00 | 0.17 | 0.10 | 1.99 |
| All | 2.36 | 0.24 | 1.09 | 3.61 | 1.76 | 0.23 | 0.57 | 2.97 | 1.27 | 0.23 | 0.10 | 2.52 |

results have been traced: T^* – minimal cycle time obtained during experiments for this instance, $B_i^A = 100(T(\pi_i^A) - T^*)/T^*$ – error of the solution obtained during i -th run of the algorithm A. Based on B_i^A for every instance and for every examined algorithm, the following parameters have been computed: AB – mean value of B_i^A , SB – standard deviation of B_i^A , $MINB$ – minimal B_i^A , $MAXB$ – maximal B_i^A . Finally, for every algorithm and every test group, average values of previously mentioned parameters have been calculated. Results are presented in Table 4. For comparison, algorithm from [7] run on the same instances provided the mean value of B_i^A as follows: $20 \times 5 - 7.61\%$, $20 \times 10 - 9.92\%$, $20 \times 20 - 9.32\%$, $50 \times 5 - 2.37\%$, $50 \times 10 - 3.41\%$, $50 \times 20 - 6.04\%$.

6 Conclusions

From Table 4 it follows that, for the considered case, redundant coding of genes linked with gene expression mechanism significantly increases efficiency of GA. Already GA-EO, which uses only first stage of the gene expression mechanism generates solutions better in average by 0.6% than GA-PMX with classic operator PMX. Further improvements of efficiency is obtainable for GA-E that uses full gene expression module - this algorithm generates solutions better by 1%. Superiority of GA-EO and GA-E over GA-PMX can be observed in all groups of instances. Moreover GA strongly dominates method from [7].

References

1. Sabena, B. Iyer, S.K.: Improved genetic algorithm for the permutation flowshop scheduling problem. *Computers and Operations Research* 31 (2004) 593–606
2. Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (1996)
3. Smutnicki, C.: Some properties of scheduling problem with storage constraints. (Polish) *Zeszyty Naukowe AGH: Automatyka* 34 (1983) 223–232
4. Grabowski, J., Pempera, J.: Sequencing of jobs in some production system. *European Journal of Operational Research* 125 (2000) 535–550
5. Leistein, R.: Flowshop sequencing with limited buffer storage. *International Journal of Production Research* 28 (1990) 2085–2100
6. Nowicki, E.: The permutation flow shop with buffers: A tabu search approach. *European Journal of Operational Research* 116 (1999) 205–219
7. McCormick, M.L., Pinedo, M.L., Shenker, S., Wolf, B.: Sequencing in an assembly line with blocking to minimize cycle time. *Operations Research* 37 (1989) 925–935
8. Burkowski, F.J.: Proximity and priority: applying a gene expression algorithm to the Travelling Salesperson Problem. *Parallel Computing* 30 (2004) 803–816
9. Goldberg, D.E., Linge, R.: Alleles, Loci and TSP. *Proceedings of the First International Conference on Genetic Algorithms*. Lawrence Erlbaum Associates, Hillsdale, NJ (1985) 154–159
10. Allahverdi, A., Aldowaisan, T.: No-wait flowshops with bicriteria of makespan and maximum lateness. *European Journal of Operational Research* 152 (2004) 132–142
11. Taillard, E.: Benchmarks for basic scheduling problems. *European Journal of Operational Research* 64 (1993) 278–285

Genetic-Greedy Hybrid Approach for Topological Active Nets Optimization

José Santos, Óscar Ibáñez,
Noelia Barreira, and Manuel G. Penedo

Computer Science Department, University of A Coruña, Spain
{santos,nbarreira,mgpenedo}@udc.es

Abstract. In this paper we propose a genetic and greedy algorithm combination for the optimization of the Topological Active Nets (TAN) model. This is a deformable model used for image segmentation that integrates features of region-based and edge-based segmentation techniques, being able to fit the edges of the objects and model their inner topology. The hybrid approach we propose can optimize the active nets through the minimization of the model energy functions and, moreover, it can provide some segmentation results unreachable by the GA method alone such as changes in the net topology.

1 Introduction

Deformable models are well-known tools for image segmentation. They were proposed by Kass et al. [1] in 1988 and applied in several fields such as segmentation, mapping or tracking and motion analysis. The active nets model was proposed by Tsumiyama and Yamamoto [2] as a variant of the deformable models that integrates features of region-based and boundary-based segmentation techniques. To this end, active nets distinguish two kind of nodes: internal nodes, related to the region-based information, and external nodes, related to the boundary-based information. The former models the inner topology of the objects whereas the latter fits the edges of the objects. The Topological Active Net (TAN) model [3] was developed as an extension of the original active net model. It solves some intrinsic problems to the deformable models such as the initialization problem. The model deformation is controlled by energy functions in such a way that the mesh energy has a minimum when the model is over the objects of the scene. This way, the segmentation process turns into a minimization task.

We have proved the superiority of a global search method by means of a Genetic Algorithm (GA) [4] in the optimization of the Topological Active Nets [5]. Our results show that the GA is less sensitive to noise than the greedy algorithm and does not depend on the parameter set or the mesh size. But the biggest limitation of the implemented GA relates to the TAN ability for performing topological changes in its structure in order to achieve a fine adjustment, detect concavities, or divide the net to segment several objects in the same image. However, the proposed hybrid approach can overcome this problem. The combination of both adaptive methods, GAs and a greedy algorithm, allows a local

search in each individual of the genetic population and in particular generations of the evolutionary process. We use a Lamarck strategy, this is, everything computationally learned in an initial genotype will be transferred to the genotype of its offspring.

There is very little work in this field with GAs, mainly in edge or surface extraction [6,7,8]. Regarding to hybrid approaches, Tohka [8] has used a GA to globally minimize the energy of a deformable surface mesh. The minimum obtained is further strengthened by a greedy algorithm. The author states that “The purpose of the GA is to perform an exploration for an approximate surface in the close vicinity of the real target. Thereafter a local search by the greedy algorithm will be more efficient (than the GA) for the surface extraction”. His results, as ours [5], show that the GA is well suited for noisy conditions and improves the energy minimization obtained by a greedy algorithm.

This paper is organized as follows. Sect. 2 introduces the basis of the TAN model. Sect. 3 explains the combination between the GA and the greedy algorithm, with emphasis in the topological changes allowed in the hybrid approach. Sect. 4 shows some results of the new method. Finally, Sect. 5 expounds the conclusions and intended future work.

2 Brief Description of Topological Active Nets

A Topological Active Net (TAN) is a discrete implementation of an elastic two dimensional mesh with interrelated nodes [3]. The model has two kinds of nodes: internal and external. Each kind of node represents different features of the objects: the external nodes fit their edges whereas the internal nodes model the internal topology of the object.

A Topological Active Net is defined parametrically as $v(r, s) = (x(r, s), y(r, s))$ where $(r, s) \in ([0, 1] \times [0, 1])$. The mesh deformations are controlled by an energy function defined as follows:

$$E(v(r, s)) = \int_0^1 \int_0^1 (E_{int}(v(r, s)) + E_{ext}(v(r, s))) dr ds \quad (1)$$

where E_{int} and E_{ext} are the internal and the external energy of the TAN, respectively. The internal energy controls the shape and the structure of the mesh whereas the external energy represents the external forces which govern the adjustment process.

The internal energy depends on first and second order derivatives which control contraction and bending, respectively. The internal energy term is defined by the following equation:

$$E_{int}(v(r, s)) = \alpha(|v_r(r, s)|^2 + |v_s(r, s)|^2) + \beta(|v_{rr}(r, s)|^2 + |v_{rs}(r, s)|^2 + |v_{ss}(r, s)|^2) \quad (2)$$

where subscripts represents partial derivatives. α and β are coefficients that control the first and second order smoothness of the net. In order to calculate

the energy, the parameter domain $[0, 1] \times [0, 1]$ is discretized as a regular grid defined by the internode spacing (k, l) and the first and second derivatives are estimated using the finite differences technique. On one hand, the first derivatives are computed using the following equations to avoid the central differences:

$$|v_r(r, s)|^2 = \frac{\|d_r^+(r, s)\|^2 + \|d_r^-(r, s)\|^2}{2} \quad |v_s(r, s)|^2 = \frac{\|d_s^+(r, s)\|^2 + \|d_s^-(r, s)\|^2}{2} \quad (3)$$

where d^+ and d^- are the forward and backward differences respectively, which are computed as follows:

$$\begin{aligned} d_r^+(r, s) &= \frac{v(r+k, s) - v(r, s)}{k} & d_r^-(r, s) &= \frac{v(r, s) - v(r-k, s)}{k} \\ d_s^+(r, s) &= \frac{v(r, s+l) - v(r, s)}{l} & d_s^-(r, s) &= \frac{v(r, s) - v(r, s-l)}{l} \end{aligned} \quad (4)$$

On the other hand, the second derivatives are estimated by:

$$\begin{aligned} v_{rr} &= \frac{v(r-k, s) - 2v(r, s) + v(r+k, s)}{k^2} & v_{ss} &= \frac{v(r, s-l) - 2v(r, s) + v(r, s+l)}{l^2} \\ v_{rs}(r, s) &= \frac{v(r-k, s) - v(r-k, s+l) - v(r, s) + v(r, s+l)}{kl} \end{aligned} \quad (5)$$

The external energy represents the features of the scene that guide the adjustment process. It is defined by the following equation:

$$E_{ext}(v(r, s)) = \omega f[I(v(r, s))] + \frac{\rho}{|\mathfrak{N}(r, s)|} \sum_{p \in \mathfrak{N}(r, s)} \frac{1}{\|v(r, s) - v(p)\|} f[I(v(p))] \quad (6)$$

where ω and ρ are weights, $I(v(r, s))$ is the intensity value of the original image in the position $v(r, s)$, $\mathfrak{N}(r, s)$ is the neighborhood of the node (r, s) and f is a function, which is different for both types of nodes since the external nodes fit the edges whereas the internal nodes model the inner features of the objects.

If the objects to detect are dark and the background is bright, the energy of an internal node will be minimum when it is on a point with a low grey level. On the other hand, the energy of an external node will be minimum when it is on a discontinuity and on a light point outside the object. In this situation, function f is defined as:

$$f[I(v(r, s))] = \begin{cases} h[\overline{I(v(r, s))}_n] & \text{for internal nodes} \\ h[I_{max} - \overline{I(v(r, s))}_n + \xi(G_{max} - G(v(r, s)))] + \delta GD(v(r, s)) & \text{for external nodes} \end{cases} \quad (7)$$

where ξ and δ are weighting terms, I_{max} and G_{max} are the maximum intensity values of image I and the gradient image G , respectively, $I(v(r, s))$ and $G(v(r, s))$ are the intensity values of the original image and the gradient image in node position $v(r, s)$, $\overline{I(v(r, s))}_n$ is the mean intensity in a $n \times n$ square and h is an appropriate scaling function. The external energy also includes the gradient distance term, $GD(v(r, s))$, this is, the distance from the position $v(r, s)$ to the nearest edge. This term introduces a continuous range in the external energy

since its value diminishes as the node gets closer to an edge. This way, the gradient distance facilitates the adjustment of the external nodes to the object boundaries.

The adjustment process consists in minimizing these energy functions. In the case of the greedy algorithm, the mesh is placed over the whole image and, in each step, the energy of each node is computed in its current position and in its nearest neighborhood. The position with the lowest energy value is selected as the new position of the node. The algorithm stops when there is no node in the mesh that can move to a position with lower energy.

3 Hybridization of the Evolutionary and Greedy Algorithms

The greedy algorithm gets good results in most cases since it takes the best local adjustment. However, the local adjustment may not be the best global one. This way, if the model reaches a wrong segmentation, it gets stuck in it. The global search provided by the GA reduces the probability of falling in local minima [5]. As the greedy algorithm, the GA was used to minimize the energy components. To this end, the genotypes coded the Cartesian coordinates of the TAN nodes. The following genetic operators were also developed or adapted to our problem:

Crossover operator. We have used an arithmetical crossover instead of the classical crossover operator because the latter produces a great number of incorrect offspring genotypes, this is, TANs with crossings in their nodes. Our alternative operator defines the new genes as a mean between the corresponding values in the two parent chromosomes.

Mutation operator. We have developed a mutation operator that avoids TAN crossings. It consists in computing the area of the 4 polygons formed by the 8 neighboring nodes and the central node that mutates. If the addition of the 4-subareas is the same before and after the mutation, the mutation is correct and it will not produce any crossing.

Spread operator. We have implemented this operator in order to maintain the diversity of sizes in the population since the proposed crossover operator tends to produce individuals with progressively similar sizes. The spread operator stretches a TAN in a given direction.

Group mutation. A group of neighboring nodes randomly selected is mutated simultaneously in the same direction and with the same value. Performing a group mutation is usually more useful than mutate only a node since the internal energy is minimum when nodes are equidistant so that, in most cases, a single mutation could not reduce the TAN energy.

Shift operator. It is used in the exploration stage and moves the net to other position in the image. This movement allows that external and internal nodes can get into the object at the same time approximately. This way, the position of the objects in the image does not affect the final node distribution.

The pure GA approach has restrictions since the TAN topology cannot be changed to obtain a better adjustment. To avoid this limitation, we have included a local greedy search stage in the evolutionary process. The idea is to perform a given number of steps of the greedy algorithm in each individual of the genetic population and in particular generations of the evolutionary process. We have followed a Lamarck strategy so all the changes made by the greedy procedure are reverted in the original genotypes.

In the hybrid approach, we have followed the principle of “first exploration and then exploitation”. This way, we have applied the greedy algorithm only in particular generations. In the beginning of the evolutionary process, the greedy algorithm is used every 15 generations. The interval of generations between the application of the greedy method decreases as the evolution progresses until a value between 1 and 5 at the end of the process.

Regarding to local search depth, this is, the number of greedy algorithm steps, different tests have induced us to use a random value between 0 and 5 or 6 (depending on the kind of image) to minimize the probability to deeply fall in local minima and, at the same time, maintain an adequate heterogeneity in the population, that can be reduced using the Lamarckian strategy.

3.1 Topological Changes: Link Cutting

The greedy algorithm can perform cuts of links between adjacent external nodes after the minimization process. The basic procedure requires the identification of the external nodes that are more distant to the object edges using the Tchebycheff’s theorem. This way, an external node v_{ext} is badly placed if its gradient distance fulfils the following inequality:

$$GD(v_{ext}) > \mu_{GD} + 3\sigma_{GD} \tag{8}$$

where μ_{GD} is the average gradient distance of the whole set of external nodes and σ_{GD} is their standard deviation.

After the identification of the outlier set, the link to remove is selected. It is the node with the highest gradient distance and its worst neighbor in the outlier set. Once the link is cut, some internal nodes become external since they are on the boundaries of the net as Fig. 1 shows. This increment of external nodes allows a better adjustment to the object boundaries.

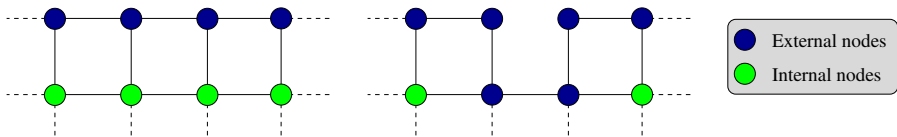


Fig. 1. Basic process in the link cutting. The figures show the TAN before and after the link cutting. After the cut, the neighboring internal nodes become external nodes.

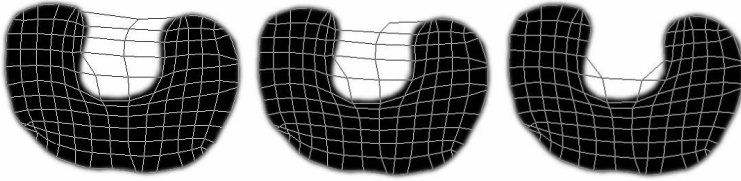


Fig. 2. Steps in the link cutting process during the local search

In the hybrid approach proposed, the local search method also uses the link cutting in order to improve the results of the pure GA approach. Fig. 2 shows three steps in the link cutting process in areas where the GA alone cannot obtain an optimal segmentation.

As we follow a Lamarck strategy, the modified topologies have to be transferred to the genotypes, so the genotype features need to be redefined. This way, for each node, the genotype must include not only the x and y coordinates, but also the directions where the node has neighbors, the kind of node (internal or external), and the priority for a link cut in the node.

Additionally, to maintain fitness coherence in the population individuals, a node that turns from internal to external after a link cut is only considered as external for fitness computation when it is over the object edges.

Crossover Operator with Different Topologies. The topological changes imply additional considerations in the crossover operator. With the previous definition, if two nets with different topologies are crossed, the resulting nets can have crossings in their connections, and the probability increases as the differences between the topologies of the selected parent nets increase. In addition, the topology of the resultant crossed nets must be decided by the operator.

Since we want populations with individuals of different topologies, a solution is to allow the crossover only between nets with the same topology. This fact implies a population categorization. Each population group with identical topology can be associated with the “niche” concept, this is, a group based on common features [4]. Only crossover between individuals of the same niche is allowed.

4 Results

We show in this section some representative examples of the improvements of the hybrid approach. In all the examples, the same image was used as the external energy for both internal and external nodes, and all the test images had 256 gray levels.

The hybrid approach and the pure GA have used a tournament selection with a window size of 3% of the population. The mutation probability was 0.0005, the crossover probability was 0.5 and the spread operator probability was set to 0.5. Regarding the *ad hoc* operators, we have experimentally set intervals of [0.001, 0.0005] and [0.005, 0.0001] for the probabilities of the shift operator and

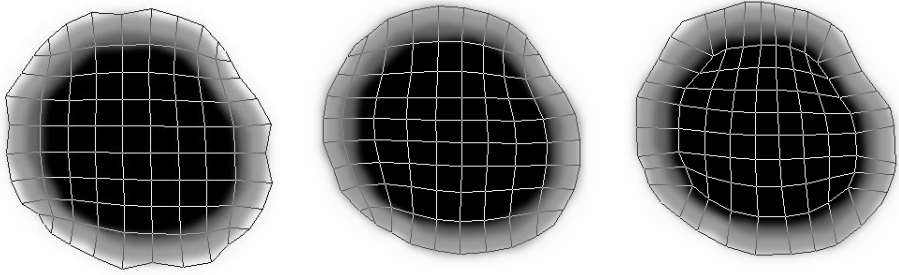


Fig. 3. Results obtained with similar mesh sizes and parameter sets in an image with fuzzy contours. The image was segmented using the greedy algorithm (left), the GA method (center), and the hybrid approach (right).

the group mutation, respectively. The examples of this section have used values in these intervals. The population is between 700–1000 individuals in the GA examples and 500 individuals in the case of the hybrid approach. Finally, we have applied between 0 and 6 steps of the greedy algorithm in the hybrid approach.

Global search advantages are clearer in images with fuzzy edges as Fig. 3 shows. In this case, only the greedy algorithm does not achieve a fine adjustment to the object edges. Table 1 summarizes the TAN parameters used in the examples.

Table 1. Summary of TAN parameters used in the segmentation examples

| <i>Method</i> | <i>Image</i> | <i>Size</i> | α | β | ω | ρ | ξ | <i>distGrad</i> |
|---------------|--------------|-------------|----------|---------|----------|--------|-------|-----------------|
| Greedy Alg. | Fig. 3 | 10 × 10 | 2.5 | 0.0001 | 1 | 2.5 | 8 | |
| | Fig. 4a | 15 × 11 | 3.5 | 0.0001 | 2 | 5.5 | 2 | |
| | Fig. 4c | 14 × 14 | 1 | 0.005 | 2 | 3 | 2 | |
| | Fig. 5 | 14 × 13 | 0.1 | 0.05 | 10 | 4.5 | 4 | |
| Genetic Alg. | Fig. 3 | 10 × 10 | 2.5 | 0.0001 | 1 | 2.5 | 8 | 6 |
| | Fig. 5 | 14 × 13 | 0.1 | 0.05 | 10 | 4.5 | 4 | 4 |
| Combination | Fig. 3 | 10 × 10 | 2.5 | 0.0001 | 1 | 2.5 | 8 | 6 |
| | Fig. 4b | 15 × 11 | 3.5 | 0.0001 | 2 | 5.5 | 2 | 6 |
| | Fig. 4d | 14 × 14 | 1 | 0.005 | 2 | 3 | 2 | 3 |
| | Fig. 5 | 14 × 13 | 0.1 | 0.05 | 10 | 4.5 | 4 | 4 |

Left column of Fig. 4 shows the main problem of the greedy search respect to the hybrid strategy. This kind of search easily falls in local minima in images with noise whereas the hybrid approach overcomes local minima thanks to the GA global search. Right column of Fig. 4 depicts another example of better adjustment of the the hybrid approach respect to the greedy algorithm.

Finally, another advantage is the hole delimitation in objects with non-uniform interior features. The aim of the TAN segmentation is to detect holes in a reliable

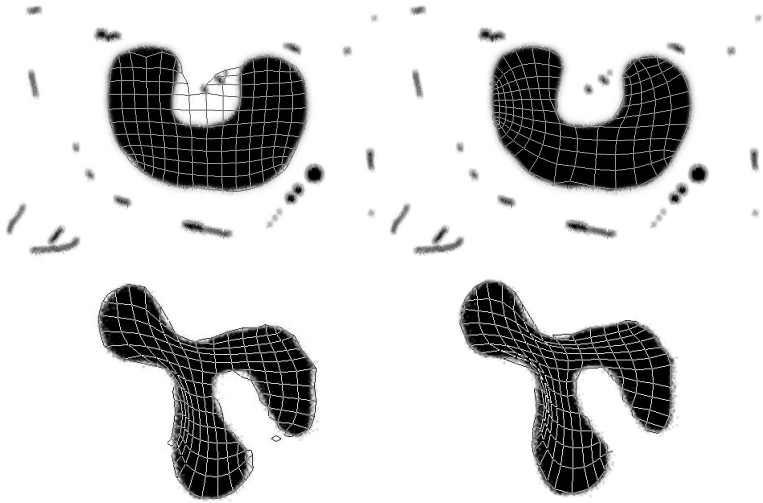


Fig. 4. Results obtained with the greedy (left) and the hybrid approach (right)

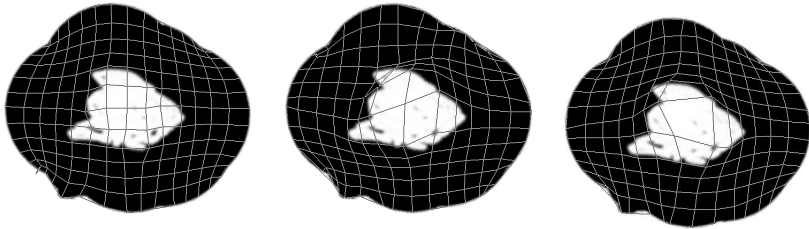


Fig. 5. Hole delimitation through internal nodes with the greedy algorithm (left), the GA (center), and the hybrid approach (right)

way, avoiding that the internal nodes fall inside them. Fig. 5 shows that the global search (pure GA and hybrid method) achieves a better hole delimitation than the greedy algorithm. The differences between local and global search increase as the hole or the mesh size increases, too.

Regarding to computation times, the greedy method is obviously faster than the GA methods whereas the hybrid approach is the slowest one. In an AMD Athlon at 1'2 GHz, the average computing time of the greedy algorithm is 5 seconds for a simple image and 30-60 seconds for a complex image, whereas the GA spends 5-10 minutes in simple images and 15-25 in complex ones. The hybrid approach requires, as average, the double as the GA, although the time depends on the number of generations and the number of iterations of the greedy search.

5 Conclusions

We present in this work a hybrid combination of a local search by means of a greedy algorithm and a global search through a genetic algorithm for the optimization of the TANs. It follows a Lamarck strategy as all the changes in the positions or nature of nodes are transferred to the original genotypes. The hybrid approach gets better results in cases where the local greedy strategy falls in local minima, such as the segmentation of fuzzy external contours or noisy images. Additionally, the hybrid approach performs a better internal segmentation.

Finally, not considered in this paper, the possibility of an automatic division of the net in several ones will allow the simultaneous segmentation of different objects in an image. These advantages justify the use of the combination in some cases, in spite of the higher computational cost.

References

1. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. *International Journal of Computer Vision* **1**(2) (1988) 321–323
2. Tsumiyama, K., Yamamoto, K.: Active net: Active net model for region extraction. *IPSP SIG notes* **89**(96) (1989) 1–8
3. Ansia, F., Penedo, M., Mariño, C., Mosquera, A.: A new approach to active nets. *Pattern Recognition and Image Analysis* **2** (1999) 76–77
4. Goldberg, D.: *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1989)
5. Ibáñez, O., Barreira, N., Santos, J., Penedo, M.: Topological active nets optimization using genetic algorithms. In: *International Conference on Image Analysis and Recognition - ICIAR06, Lecture Notes in Computer Science*. Volume 4141. (2006) 272–282
6. Ballerini, L.: Medical image segmentation using genetic snakes. In: *Proceedings of SPIE: Application and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation II*. Volume 3812. (1999) 13–23
7. Fan, Y., Jiang, T., Evans, D.: Volumetric segmentation of brain images using parallel genetic algorithm. *IEEE Tran. on Medical Imaging* **21**(8) (2002) 904–909
8. Tohka, J.: Global optimization of deformable surface meshes based on genetic algorithms. In: *Proceedings ICIAP, IEEE Computer Society* (2001) 459–464

On Sum Coloring of Graphs with Parallel Genetic Algorithms

Zbigniew Kokosiński^{1,2} and Krzysztof Kwarciany¹

¹ Cracow University of Technology, Faculty of Electrical & Computer Eng.,
ul. Warszawska 24, 31-155 Kraków, Poland

zk@pk.edu.pl

² Tischner European University, ul. Westerplatte 11, 31-033 Kraków, Poland

Abstract. Chromatic number, chromatic sum and chromatic sum number are important graph coloring characteristics. The paper proves that a parallel metaheuristic like the parallel genetic algorithm (PGA) can be efficiently used for computing approximate sum colorings and finding upper bounds for chromatic sums and chromatic sum numbers for hard-to-color graphs. Suboptimal sum coloring with PGA gives usually much closer bounds than theoretical formulas known from the literature.

1 Introduction

Graph Coloring Problem (GCP) and Graph Chromatic Sum problem (GCS) belong to the class of NP-hard combinatorial optimizations problems [6,12].

GCP is defined for an undirected graph $G(V, E)$ as an assignment of available colors $\{1, \dots, k\}$ to graph vertices providing that adjacent vertices receive different colors and the number of colors k is minimal. The resulting coloring is called conflict-free and k is called graph chromatic number $\chi(G)$.

GCS is defined for an undirected graph $G(V, E)$ as an assignment of available colors $\{1, \dots, h\}$ to graph vertices providing that sum of all color numbers in a conflict-free coloring must be minimal. The minimum number of colors h in a minimum-sum coloring is called chromatic sum number $s(G)$, $s(G) \geq \chi(G)$.

A generalization of GCS is the Minimum Sum Multicoloring problem related to distributed resource allocation [1].

Intensive research has been conducted in the area of graph coloring and resulted in a large number of exact and approximate algorithms, heuristics and metaheuristics. The Graph Coloring Problem was the subject of Second DIMACS Implementation Challenge in 1993 and Computational Symposium on Graph Coloring and Generalizations in 2002. A collection of hard to color graph instances in DIMACS format and summary of results are available at [16,17].

Chromatic number and chromatic sum number are important coloring parameters that characterize graphs. While chromatic numbers for most DIMACS graphs are determined little is known about their chromatic sums. Some theoretical lower and upper bounds for general and some specific graphs are reported

in the literature [14]. In addition, new lower bounds for chromatic sums are established in the present paper. However, in most cases the gap between bounds is still large.

The objective of this paper is to compute experimentally sum colorings, closer upper bounds for chromatic sums and evaluate their corresponding approximate chromatic sum numbers $s(G)$ for a set of DIMACS graphs. For computations the parallel genetic algorithm (PGA) [4] is adapted which has already been found to be efficient in solving GCP problem [10,11]. For sum coloring another definition of the fitness function is applied. As a result of conducted computations the theoretical upper bounds for chromatic sums of the selected benchmark graphs are significantly improved by our experimental data.

The rest of the paper is organized as follows. The next section presents basic definitions and theoretical bounds on the graph chromatic sum and the chromatic sum number. Next, the migration model of PGA is characterized in section 3. Some specific operators and functions used in the PGA are described in section 4. Sections 5 and 6 contain main computational results and conclusions.

2 Graph Coloring Sum Problem – Definitions and Bounds

Let us define formally the optimization problems GCP and GCS.

For a given graph $G(V, E)$, where V – set of graph vertices, $|V| = n$, and E – set of graph edges, $|E| = m$, the optimization problem GCP is formulated as follows: find the minimum positive integer k , $k \leq n$, and a function $c : V \rightarrow A$, $A = \{1, \dots, k\}$, such that $c(u) \neq c(v)$ whenever $(u, v) \in E$. The obtained value of k is referred to as graph chromatic number $\chi(G)$.

The cost of a vertex coloring c is the sum

$$\sum_{v \in V} c(v). \quad (1)$$

The chromatic sum of graph G is

$$\sum(G) = \min_c \sum_{v \in V} c(v). \quad (2)$$

The optimization problem GCS is formulated as follows: for a given graph $G(V, E)$ find the minimum positive integer h , $\chi(G) \leq h \leq n$, and a function $c : V \rightarrow B$, $B = \{1, \dots, h\}$ such that $\sum(G)$ is minimal. The obtained value of h is referred to as graph sum chromatic number $s(G)$.

In the above definitions the color set $A = \{1, \dots, k\}$ is an interval subsets of \mathbf{N} , i.e. $A = \{j \in \mathbf{N} : 1 \leq j \leq k\}$. Similarly, the color set $B = \{1, \dots, h\}$ is an interval subsets of \mathbf{N} .

A conflict-free vertex coloring c is a partition of V into independent sets C_i , where i is a color number in a color set. Thus, an optimal sum coloring minimizes

$$\sum_{i \in B} i \cdot |C_i|. \quad (3)$$

For any optimal sum coloring c of G the following properties hold:

$$|C_1| \geq \dots |C_i| \geq \dots \geq |C_h|, \tag{4}$$

and

$$\forall_{i < j} \forall_{v \in C_j} \exists_{u \in C_i} \{u, v\} \in E, \tag{5}$$

where i, j are color values.

Parameters $\chi(G)$ and $s(G)$ are related exclusively to the given graph $G(V, E)$ and are invariable with respect to the formulation of problems GCP and GCS.

An exemplary graph $G(V, E)$ with ten vertices is shown in Fig.1.

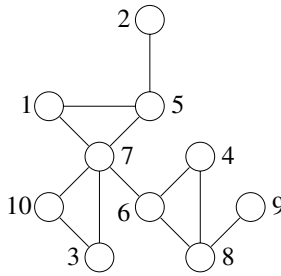


Fig. 1. An exemplary graph $G(V, E)$

In graph coloring problems k -colorings of graph vertices are encoded in chromosomes representing set partitions with exactly k blocks. There are two equivalent notations for vertex colorings that are commonly used in the algorithm design.

In *assignment* representation available colors are assigned to an ordered sequence of graph vertices. Thus, the vector $c = \langle c[1], c[2], \dots, c[n] \rangle$ represents a vertex coloring. For the graph in Fig.1, an optimal 3-coloring is denoted by the vector $c = \langle 1, 2, 3, 2, 3, 1, 2, 3, 2, 1 \rangle$.

In *partition* representation the vertex coloring is a unique sequence of partition blocks. Each block of the partition p does correspond to a single color. The elements within each partition block are ordered in the increasing lexicographic order, and all blocks are ordered increasingly according to the value of their first elements. For our graph the same optimal 3-coloring is denoted by the partition $p = \{1, 6, 10\}\{2, 4, 7, 9\}\{3, 5, 8\}$.

The cost of the above optimal 3-colorings is $\sum_{v \in V} c(v) = 20$.

In some cases of non-optimal colorings the result can be improved by re-ordering of partition blocks according to decreasing order of their sizes and assigning them increasing color numbers according to property 4. In this way the coloring presented above can be improved to $c = \langle 2, 1, 3, 1, 3, 2, 1, 3, 1, 2 \rangle$ or $p = \{2, 4, 7, 9\}\{1, 6, 10\}\{3, 5, 8\}$ and the resulting chromatic sum is $\sum_{v \in V} c(v) = 19$.

The minimum chromatic sum can be obtained when the number of colors is not minimal.

A number of theoretical lower and upper bounds on chromatic sum $\sum(G)$ with the color set $B = \{1, \dots, s(G)\}$ as well as bounds on the chromatic sum number $s(G)$ is known from the literature [12].

The chromatic sum $\sum(G)$ is bounded by:

$$\begin{aligned} n &\leq \sum(G) & (6) & \sum(G) \leq n + m & (9) \\ \lceil \sqrt{8m} \rceil &\leq \sum(G) & (7) & \sum(G) \leq \lfloor 3(m + 1)/2 \rfloor & (10) \\ & & & \sum(G) \leq n(\chi(G) + 1)/2 & (11) \end{aligned}$$

The chromatic sum number $s(G)$ is bounded by:

$$\begin{aligned} \chi(G) &\leq s(G) & (12) & s(G) \leq \lfloor \sqrt{n(\chi(G) + 1)} \rfloor & (13) \\ & & & s(G) \leq \Delta(G), & (14) \end{aligned}$$

where $\Delta(G)$ denotes degree of the graph G , i.e. maximum degree of $v \in V$. (Remark: for full graphs and odd cycles $s(G) = \Delta(G) + 1$).

We propose two new theoretical lower bounds according to the following theorem:

Theorem 1. *The graph chromatic sum defined above satisfies the following two inequalities:*

$$n + s(G)(s(G) - 1)/2 \leq \sum(G) \tag{15}$$

$$n + \chi(G)(\chi(G) - 1)/2 \leq \sum(G) \tag{16}$$

Proof. In order to prove [15] it is sufficient to show that the cost of optimal sum coloring with exactly $h = s(G)$ colors is at least : $\sum_{i=1}^{s(G)} i + (n - s(G)) = s(G)(s(G) + 1)/2 + (n - s(G)) = n + s(G)(s(G) - 1)/2$. The inequality [16] follows directly from [12] and [15].

Bounds [11], [12], [13] and [16] are as hard to compute as $\chi(G)$, but for many benchmark graphs $\chi(G)$ is known.

3 Migration Model of Parallel Genetic Algorithms

There are many models of parallelism in evolutionary algorithms: master–slave PGA, migration based PGA, diffusion based PGA, PGA with overlapping subpopulations, population learning algorithm, hybrid models etc.

The migration model used in this paper is basically the same as the model used in [10],[11]. Migration models of PGAs consist of a finite number of disjoint subpopulations that evolve in parallel on their "islands" and exchange genetic informations under control of a migration operator. Co–evolving subpopulations

are built of individuals of the same type and are ruled by one adaptation function. The selection process is decentralized.

During the migration phase every island sends its representatives to all other islands and receives the representatives from all co-evolving subpopulations. The migration process is fully characterized by migration size, distance between populations and migration scheme. Migration size determines the emigrant fraction of each population. This parameter is limited by capacity of islands to accept immigrants. The distance between migrations determines how often the migration phase of the algorithm occurs. Three migration schemes are applied: no migration, migration of randomly selected individuals and migration of best individuals of the subpopulation. In our algorithm a specific model of migration is applied in which islands use two copies of genetic information: migrating individuals still remain members of their original subpopulation. In other words they receive new "membership" without losing the former one. Incoming individuals replace the chromosomes of host subpopulation at random. Then, a selection process is performed. The rationale behind such a model is as follows. Even if the best chromosomes of host subpopulation are eliminated they shall survive on other islands where their copies were sent. On the other hand any elitist scheme or preselection applied to the replacement phase leads to premature elimination of worse individuals and lowers the overall diversity of subpopulation.

Computer experiments provide an evidence that parallel genetic algorithms can be efficiently used for a class of graph coloring problems [10,11].

4 Genetic Operators for GCS

In this section a collection of genetic crossover, mutation and selection operators is introduced that is used in our PGA. Two recombination operators: CEX, GPX and the mutation operator First Fit were initially designed for GCP (for more details see [10,11]). The cost function and selection operator is designed especially for GCS.

4.1 Recombination Operators

In conflict-based crossovers for GCP the assignment representation of colorings is used and the offspring tries to copy conflict-free colors from their parents. The recombination operator called Conflict Elimination Crossover (CEX) reveals some similarity to the classical crossover. Each parental chromosome p and r is partitioned into two blocks. The first block consists of conflict-free nodes while the second one is built of the remaining nodes that break the coloring rules. This second block in both chromosomes is then replaced by corresponding colors taken from the other parent. This recombination scheme provides inheritance of all good properties of one parent and gives the second parent a chance to reduce the number of existing conflicts. However, if a chromosome represents a feasible coloring the recombination mechanism will not work properly. Therefore, the

recombination must be combined with an efficient mutation mechanism. As a result two chromosomes s and t are produced. The operator CEX is as simple and easy to implement as the classical crossover.

4.2 Greedy Partition Crossover

The method called Greedy Partition Crossover (GPX) was designed by Galinier and Hao for recombination of colorings or partial colorings in partition representation [5]. It is assumed that both parents are randomly selected partitions with exactly k blocks that are independent sets. The result is a single offspring (a coloring or a partial coloring) that is built successively in a greedy way. In each odd step select the maximum block from the first parent is selected. Then the block is added to the result and all its nodes from the both parents are removed. In each even step the maximum block is selected from the second parent. Then the block is added to the result and all its nodes from the both parents are removed. The procedure is repeated at most k times since in some cases the offspring has less blocks then the parents. This possibility is not considered in the original paper [5]. Finally, unassigned vertices (if they exist) are assigned at random to existing blocks of partition. The first parent is replaced by the offspring while the second parent is returned to population and can be recombined again in the same generation.

4.3 Mutation Operator

The mutation operation called First Fit (FF) is designed for colorings in partition representation and is well suited for GCS. In this mutation one block of the partition is selected at random and we try to make a conflict-free assignment of its vertices to other blocks using the heuristic First Fit. Vertices with no conflict-free assignment remain in the original block. Thus, as a result of the mutation First Fit the color assignment is partially rearranged and the number of partition blocks is often reduced by one.

4.4 Selection

Selection process maintains constant size of population selected by means of a fitness function.

The quality of a solution is measured by the following cost function:

$$f(c) = \sum(G) + \sum_{(u,v) \in E} q(u, v) + d, \text{ where:}$$

c – is a graph coloring,

$\sum(G) = \sum_{v \in V} c(v)$ – is the sum of colors used in c ,

q – is a penalty function for pairs of vertices connected by an edge $(u, v) \in E$:

$$q(u, v) = \begin{cases} 5, & \text{when } c(u) = c(v) \\ 0, & \text{otherwise} \end{cases}$$

d – is a general penalty function applied to graph colorings c :

$$d = \begin{cases} 100, & \text{when } \sum_{(u,v) \in E} q(u, v) > 0 \\ 0, & \text{when } \sum_{(u,v) \in E} q(u, v) = 0. \end{cases}$$

In many cases less colors results in more conflicts. Modeling the cost function we can favour conflict-free colorings by setting values of $q(u, v)$ and d . On the other hand conflict colorings with less colors can also be useful. Therefore, we decided to set relatively low values of $q(u, v)$ and d . Thus, with the cost function given above, all k' -colorings with i conflicts, $k' \leq (k - 2i - 2)$, are better then conflict-free k -colorings.

The proportional (roulette) selection is performed with the fitness function $1/f(c)$.

In section 5 the theoretical bounds for selected subset of DIMACS graphs are given and compared with the bounds computed experimentally.

5 Experimental Results

Initial experiments for graph GCS problem were conducted with greedy coloring heuristic GIS and conventional genetic algorithm.

GIS is an approximation algorithm proposed by Johnson [8] which computes valid graph coloring via computing subsequent approximate independent sets (ISs). Decreasing ordering the graph ISs according to their powers and assigning to them increasing color numbers from the color set is one of the methods of finding approximate minimal sum coloring in graphs [14]. For each selected graph the program implementing GIS heuristic was executed once.

The best sum 3-coloring of the graph G from Fig.1 computed by GIS heuristic was the partition $p = \{1, 2, 3, 4, 9\}\{5, 6, 10\}\{7, 8\}$ with the coloring cost $\sum_{v \in V} c(v) = 17$.

Genetic algorithms (GA) are metaheuristics often used for GCP [3,5]. GA was applied with coloring chromosomes in assignment representation, standard 1-point crossover, 1-point mutation and proportional selection. The initial population was generated at random. We set the following parameters of GA: *population size* = 200 , *crossover probability*= 0.8, *mutation probability* = 0.1. All experiments were repeated several times for each selected graph.

The best sum 3-coloring of the graph G from Fig.1 obtained by GA was $c = \langle 1, 1, 2, 2, 2, 1, 3, 3, 1, 1 \rangle$ with cost $\sum_{v \in V} c(v) = 17$.

Further experiments were conducted with migration model of PGA described in section 3.

In this paper PGA is applied with coloring chromosomes in both assignment and partition representations, CEX and GPX crossovers, respectively, First Fit mutation and proportional selection. The initial population was generated at random. We used the following parameters of PGA: *population size* = 60 , *number of islands* = 3 or 5, *crossover probability*= 0.8, *mutation probability* = 0.1 with

GPX and 0.5 with CEX, *number of iterations* = 5000 with GPX and 10000 with CEX. All experiments were repeated several times for each selected graph. In some cases PGA generates colorings not satisfying property 4. Therefore, the final reordering of assigned colors was necessary for improving the value of graph chromatic sum.

Preliminary experiments on graph coloring benchmarks showed us that PGA for GCS outperforms both the GIS heuristic and the conventional GA. The comparison results are not presented in the present paper.

All our experiments with PGA were performed on a computer with Pentium 4 processor (3,06GHz, 1GB RAM). The programs generated detailed reports and basic statistics. Processing times are not reported in the present paper.

We used the graph coloring instances available in the web archive [16]. This is a collection of graphs in DIMACS format with known parameters, including graph chromatic numbers. In the preprocessing phase we converted list of edges representation into adjacency matrix representation.

The theoretical bounds and the best computational results for 16 DIMACS benchmarks with at most 200 vertices are presented in Table 1. We used the following notations: L.B. and U.B. stand for lower and upper bound, respectively, A.B. denotes approximate upper bound for $s(G)$ (in fact the exact value of $s(G)$ can be lower then, equal or greater then A.B., satisfying bounds 12–14).

All theoretical bounds on $\sum(G)$ were improved except L.B. and U.B for queen5.5 and U.B for queen7.7. Also approximate upper bounds for $s(G)$ are significantly lower in comparison to theoretical upper bounds.

Table 1. Bounds on chromatic sums and chromatic sum numbers

| $G(V, E)$ | n | m | $\chi(G)$ | $\Delta(G)$ | theoretical bounds | | | | | | experimental bounds | |
|-----------|-----|------|-----------|-------------|--------------------|------|------|------|--------|------|---------------------|--------|
| | | | | | $\sum(G)$ | | | | $s(G)$ | | $\sum(G)$ | $s(G)$ |
| | | | | | L.B. | rule | U.B. | rule | U.B. | rule | U.B. | A.B. |
| anna | 138 | 493 | 11 | 71 | 193 | 16 | 631 | 9 | 38 | 13 | 281 | 11 |
| david | 87 | 406 | 11 | 82 | 142 | 16 | 494 | 9 | 30 | 13 | 243 | 11 |
| huck | 74 | 301 | 11 | 53 | 129 | 16 | 375 | 9 | 28 | 13 | 243 | 11 |
| jean | 80 | 254 | 10 | 33 | 125 | 16 | 334 | 9 | 28 | 13 | 218 | 10 |
| queen5.5 | 25 | 160 | 5 | 12 | 36 | 7 | 75 | 11 | 11 | 13 | 75 | 5 |
| queen6.6 | 36 | 290 | 7 | 19 | 57 | 16 | 144 | 11 | 15 | 13 | 138 | 8 |
| queen7.7 | 49 | 476 | 7 | 24 | 70 | 16 | 196 | 11 | 18 | 13 | 196 | 7 |
| queen8.8 | 64 | 728 | 9 | 27 | 100 | 16 | 320 | 11 | 24 | 13 | 302 | 10 |
| games120 | 120 | 638 | 9 | 13 | 156 | 16 | 600 | 11 | 13 | 14 | 460 | 9 |
| miles250 | 128 | 387 | 8 | 16 | 156 | 16 | 515 | 9 | 16 | 14 | 347 | 8 |
| miles500 | 128 | 1170 | 20 | 38 | 318 | 16 | 1298 | 9 | 38 | 14 | 762 | 20 |
| myciel3 | 11 | 20 | 4 | 5 | 17 | 16 | 27 | 11 | 5 | 14 | 21 | 4 |
| myciel4 | 23 | 71 | 5 | 11 | 33 | 16 | 69 | 11 | 10 | 13 | 45 | 5 |
| myciel5 | 47 | 236 | 6 | 23 | 62 | 16 | 164 | 11 | 16 | 13 | 93 | 6 |
| myciel6 | 95 | 755 | 7 | 47 | 116 | 16 | 380 | 11 | 27 | 13 | 189 | 7 |
| myciel7 | 191 | 2360 | 8 | 95 | 219 | 16 | 859 | 11 | 39 | 13 | 382 | 8 |

6 Conclusions

The paper shows that general heuristics like PGA can be used for finding approximate solutions to the optimization problem GCS.

The theoretical and experimental results obtained in this paper provide a better insight into the chromatic sum problem by improving known lower and upper bounds on chromatic sums $\sum(G)$ and, simultaneously, by a closer approximation of chromatic sum numbers $s(G)$ for selected benchmark graphs from DIMACS web archive. The authors hope that further progress in computing bounds can be obtained with the help of parallel metaheuristics [2]. The presented results can be useful reference data in future research in this area. Computing exact chromatic sums $\sum(G)$ for all DIMACS benchmarks remains an open question.

References

1. Bar-Noy A., Bellare M., Halldórsson M.M., Shachnai H., Tamir T.: On chromatic sums and distributed resource allocation, *Inform. and Comput.* 140 (1998) 183–202
2. Crainic T.G., Nourredine, H.: *Parallel Meta-Heuristics Applications*, [in:] *Parallel Metaheuristics*, E. Alba (Ed.), John Wiley & Sons (2005)
3. Croitoriu C., Luchian H., Gheorghies O., Apetrei A.: A new genetic graph coloring heuristic, *Computational Symposium on Graph Coloring and Generalizations COLOR'02*. [in:] *Constraint Programming, Proc. of the Int. Conf. CP'02* (2002)
4. Cantú-Paz, E.: *Efficient and accurate parallel genetic algorithms*, Kluwer (2000)
5. Galinier P., Hao J-K.: *Hybrid evolutionary algorithms for graph coloring*, *J. Combinatorial Optimization* (1999) 374–397
6. Garey R., Johnson D.S.: *Computers and intractability. A guide to the theory of NP-completeness*, Freeman (1979)
7. Jensen T.R., Toft B.: *Graph coloring problems*, Wiley Interscience (1995)
8. Johnson D.S.: *Approximation algorithms for combinatorial problems*, *J. Computer System Science*, 9 (1974) 256–278
9. Johnson D.S., Trick M.A.: *Cliques, coloring and satisfiability: Second DIMACS Implementation Challenge*, DIMACS Series in Discr. Math. and Theor. Comp. Sc. Vol.26 (1996)
10. Kokosiński Z., Kolodziej M., Kwarcianny K.: *Parallel genetic algorithm for graph coloring problem*, *Proc. ICCS'2004, LNCS 3036* (2004) 215–222
11. Kokosiński Z., Kwarcianny K., Kolodziej M.: *Efficient graph coloring with parallel genetic algorithms*, *Computing and Informatics*, 24 (2005) 123–148
12. Kubicka E., Schwenk A.J.: *An introduction to chromatic sums*, *Proc. 17th Annual AcM Computer Science Conf.* (1989) 39–45
13. Kubicka E., Kubicki G., Kountanis D.: *Approximation algorithms for the chromatic sum*, *LNCS 507* (1991) 15–21
14. Małafiejski M.: *Sum Coloring of Graphs*. [in:] Kubale M. (Ed.): *Graph Colorings*, American Math. Society, *Contemporary Mathematics Vol. 352* (2004) 55–65
15. de Werra D.: *Heuristics for graph coloring*, [in:] Tinhofer G. et al. (eds.) *Computational graph theory*, Springer-Verlag (1990) 191–208
16. <http://mat.gsia.cmu.edu/COLOR/instances.html>
17. <http://mat.gsia.cmu.edu/COLORING03/>

Liquid State Genetic Programming

Mihai Oltean

Department of Computer Science
Faculty of Mathematics and Computer Science
Babeş-Bolyai University, Kogălniceanu 1
Cluj-Napoca, 3400, Romania
moltean@cs.ubbcluj.ro

Abstract. A new Genetic Programming variant called Liquid State Genetic Programming (LSGP) is proposed in this paper. LSGP is a hybrid method combining a dynamic memory for storing the inputs (the liquid) and a Genetic Programming technique used for the problem solving part. Several numerical experiments with LSGP are performed by using several benchmarking problems. Numerical experiments show that LSGP performs similarly and sometimes even better than standard Genetic Programming for the considered test problems.

1 Introduction

Liquid State Machine (LSM) is a technique recently described in the literature [3], [5]. It provides a theoretical framework for a neural network that can process signals presented temporally. The system is comprised of two subcomponents, the *liquid* and the *readout*. The former acts as a decaying memory, while the latter acts as the main pattern recognition unit.

Liquid State Genetic Programming (LSGP), introduced in this paper, is similar to both LSM and Genetic Programming (GP) [1] as it uses a dynamic memory (the liquid) and a GP algorithm which is the actual problem solver. The liquid is simulated by using some operations performed on the inputs. The purpose of the liquid is to transform the inputs into a form which can be more easily processed by the problem solver (GP). The liquid acts as some kind of preprocessor which combines the inputs using the standard functions available for the internal nodes of GP trees.

We have applied the LSGP on several test problems. Due to the space limitation we will present the results only for one difficult problem: even-parity. We choose to apply the proposed LSGP technique to the even-parity problems because according to Koza [1] these problems appear to be the most difficult Boolean functions to be detected via a blind random search.

Evolutionary techniques have been extensively used for evolving digital circuits [1], [4], due to their practical importance. The case of even-parity circuits was deeply analyzed [1], [4] due to their simple representation. Standard GP was able to solve up to even-5 parity [1]. Using the proposed LSGP we are able to easily solve up to even-8 parity problem.

Numerical experiments, performed in this paper, show that LSGP performs similarly and sometimes even better than standard GP for the considered test problems.

The paper is organized as follows: Liquid State Machines are briefly described in Sect. 2. In Sect. 3 the proposed Liquid State Genetic Programming technique is described. The results of the numerical experiments for are presented in Sect. 4.2. Conclusions and further work directions are given in Sect. 6.

2 Liquid State Machines – A Brief Introduction

Liquid computing is a technique recently described in the literature [3], [5]. It provides a theoretical framework for an Artificial Neural Network (ANN) that can process signals presented temporally. The system is comprised of two sub-components, the *liquid* and the *readout*. The former acts as a decaying memory, while the latter acts as the main pattern recognition unit. To understand the system, a simple analogy is used. If a small rock thrown in a cup (pool) of water it will generate ripples that are mathematically related both to characteristics of the rock, as well as characteristics of the pool at the moment that the rock was thrown in. A camera takes still images of the water's ripple patterns. A computer then analyzes these still pictures. The result is that the computer should know something about the rock that was thrown in. For example, it should know about how long ago the rock was thrown. The rock represents a single bit from an input stream, or an action potential. The water is the liquid memory. The computer functions as the readout.

Translated into ANN's language the idea behind Liquid State Machines has been implemented [3], [5] as follows: Two ANNs are used: one of them plays the role of the liquid and the other is the actual solver. The inputs of the first network are the problem inputs and this network will reshape (modify) the inputs in order to be more easily handled by the second network. This second network, which is the actual problem solver, takes the inputs from some of the nodes (randomly chosen) of the first network. In this way, it is hoped that the structure of the second network (the actual problem solver) is simpler than the case when a single network is used for solving the problem.

3 Liquid State Genetic Programming

In this section the proposed LSGP technique is described. LSGP is a hybrid method combining a technique for manipulating the liquid and a GP technique for the individuals.

For a better understanding we will start with a short example on how a LSGP individual looks like for the even-parity problem. The example is depicted in Fig. 1.

The liquid can be viewed as a set (a pool) of items (or individuals) which are subject to some strict rules which will be deeply explained in the next sections.

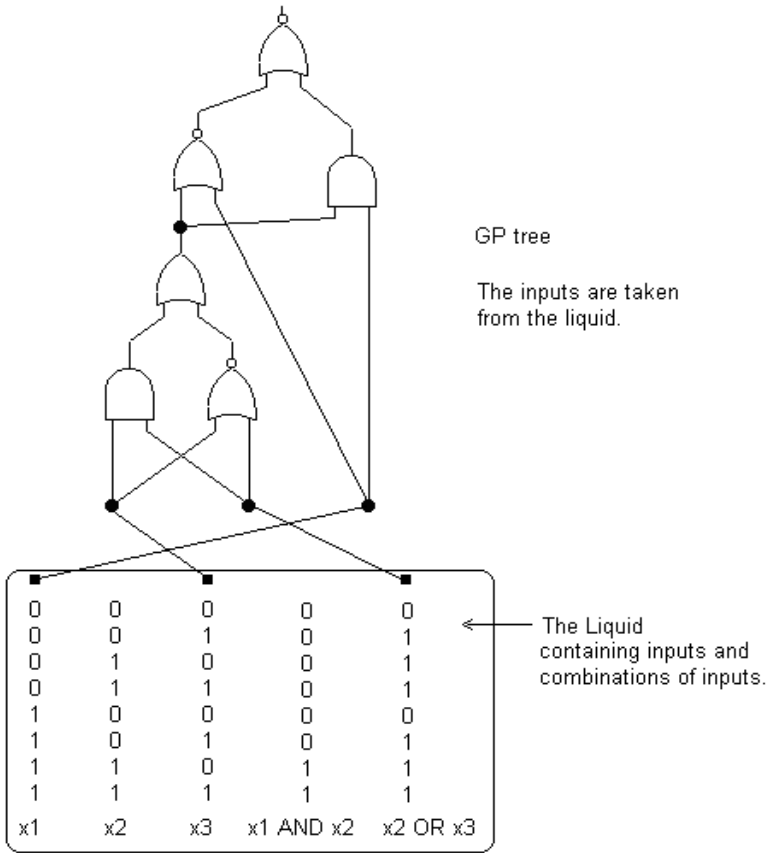


Fig. 1. A *liquid* and a GP program for the even-3-parity problem. The *liquid* contains 5 items. Three of them are the standard inputs for the even-3-parity problem and 2 of them are combinations of the standard inputs. The inputs for the GP program are taken from the *liquid*. The gates used by the GP program are the standard one: AND, OR, NAND, NOR.

The liquid is simulated by using some operations performed on the inputs. The purpose of the *liquid* is to transform the inputs into a form which can be more easily processed by the problem solver (GP). The liquid acts as some kind of preprocessor which combines the inputs using the standard functions available for the internal nodes of GP trees.

The liquid and its accompanying rules can also be viewed as a simple GP algorithm that manipulates only the output of the tree rather than the entire tree. The state of the liquid will also evolve during the search process.

The GP algorithm is a standard one [1] and due to the space limitations will not be detailed in this paper.

3.1 Prerequisite

The quality of a GP individual is usually computed using a set of fitness cases [1]. For instance, the aim of symbolic regression is to find a mathematical expression that satisfies a set of m fitness cases.

We consider a problem with n inputs: x_1, x_2, \dots, x_n and one output f . The inputs are also called terminals [1]. The function symbols that we use for constructing a mathematical expression are $F = \{+, -, *, /, \sin\}$.

Each fitness case is given as an array of $(n + 1)$ real values:

$$v_1^k, v_2^k, v_3^k, \dots, v_n^k, f_k$$

where v_j^k is the value of the j^{th} attribute (which is x_j) in the k^{th} fitness case and f_k is the output for the k^{th} fitness case.

Usually more fitness cases are given (denoted by m) and the task is to find the expression that best satisfies all these fitness cases. This is usually done by minimizing the quantity:

$$Q = \sum_{k=1}^m |f_k - o_k|,$$

where f_k is the target value for the k^{th} fitness case and o_k is the actual (obtained) value for the k^{th} fitness case.

3.2 Representation of Liquid's Items

Each individual (or item) in the liquid represents a mathematical expression obtained so far, but this individual does not explicitly store this expression. Each individual in the liquid stores only the obtained value, so far, for each fitness case. Thus an individual in the liquid is an array of values:

$$(o_1, o_2, o_3, \dots, o_m)^T,$$

where o_k is the current value for the k^{th} fitness case and $()^T$ is the notation for the transposed array. Each position in this array (a value o_k) is a gene. As we said it before behind these values is a mathematical expression whose evaluation has generated these values. However, we do not store this expression. We store only the values o_k .

3.3 Initial Liquid

The initial liquid contains individuals (items) whose values have been generated by simple expressions (made up by a single terminal). For instance, if an individual in the initial liquid represents the expression:

$$E = x_1,$$

then the corresponding individual in the liquid is represented as:

$$C = (v_1^1, v_1^2, v_1^3, \dots, v_1^m)^T$$

where v_j^k has been previously explained.

Example 1. For the particular case of the even-3-parity problem we have 3 inputs x_1, x_2, x_3 (see Fig. 1) and $2^3 = 8$ fitness cases which are listed in Table 1:

Table 1. The truth table for the even-3-parity problem

| x_1 | x_2 | x_3 | Output |
|-------|-------|-------|--------|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

Each item in the liquid is an array with 8 values (one for each fitness case). There are only 3 possible items for initial liquid: $(00001111)^T$, $(00110011)^T$ and $(01010101)^T$ corresponding to the values for variables x_1, x_2 and x_3 . Other items can appear in the liquid later as effect of the specific genetic operators (see Sect. 3.4). Of course, multiple copies of the same item are allowed in a liquid at any moment of time.

It is desirable, but not necessary to have each variable represented at least once in the initial liquid. This means that the number of items in the liquid should be larger than the number of inputs of the problem being solved. However, an input can be inserted later as effect of the insertion operator (see Sect. 3.4).

3.4 Operators Utilized for Modifying the Liquid

In this section the operators used for the Liquid part of the LSGP are described. Two operators are used: combination and insertion. These operators are specially designed for the liquid part of the proposed LSGP technique.

Recombination. The recombination operator is the only variation operator that creates new items in the liquid. For recombination several items (the parents) and a function symbol are selected. The offspring is obtained by applying the selected operator for each of the symbols of the parents.

The number of parents selected for combination depends on the number of arguments required by the selected function symbol. Two parents have to be selected for combination if the function symbol is a binary operator. A single parent needs to be selected if the function symbol is a unary operator.

Example 2. Let us suppose that the operator AND is selected. In this case two parents (items in the liquid):

$$C_1 = (p_1, p_2, \dots, p_m)^T \text{ and}$$

$$C_2 = (q_1, q_2, \dots, q_m)^T$$

are selected and the offspring O is obtained as follows:

$$O = (p_1 \text{AND} q_1, p_2 \text{AND} q_2, \dots, p_m \text{AND} q_m)^T.$$

Example 3. Let us suppose that the operator NOT is selected. In this case one parent (item in the liquid):

$$C_1 = (p_1, p_2, \dots, p_m)^T$$

is selected and the offspring O is obtained as follows:

$$O = (\text{NOT}(p_1), \text{NOT}(p_2), \dots, \text{NOT}(p_m))^T.$$

Remark 1. The operators used for combining genes of the items in the liquid must be restricted to those used by the main GP algorithm. For instance, if the function set allowed in the main GP program is $F = \{\text{AND}, \text{OR}\}$, then for the recombination part of the liquid we can use only these 2 operators. We cannot use other functions such as NOT, XOR etc.

Insertion. This operator inserts a simple expression (made up of a single terminal) in the liquid. This operator is useful when the liquid contains items representing very complex expressions that cannot improve the search. By inserting simple expressions we give a chance to the evolutionary process to choose another direction for evolution.

3.5 The LSGP Algorithm

Due to the special representation and due to the special operators, LSGP uses a special generational algorithm which is given below.

The LSGP algorithm starts by creating a random population of GP individuals and a random liquid. The evolutionary process is run for a fixed number of generations. The underlying algorithm for GP has been deeply described in [1].

The modifications in liquid are also generation-based and usually they have a different rate compared to modifications performed for the GP individuals. From the numerical experiments we have deduced that the modifications in the liquid should not occur as often as the modifications within GP individuals. Thus an update of the liquid's items will be performed only after 5 generations of the GP algorithm.

The updates for the liquid are as follows: at each generation of the liquid the following steps are repeated until the new *LiquidSize* items are obtained: with a probability p_{insert} generate an offspring made up of a single terminal (see the Insertion operator, Sect. 3.4). With a probability $1 - p_{\text{insert}}$ randomly select two parents. The parents are recombined in order to obtain an offspring (see Sect. 3.4). The offspring enters the liquid of the next generation.

A basic form of elitism is employed by the GP algorithm: the best so far GP individual along with the current state of the liquid is saved at each generation during the search process. The best individual will provide solution of the problem.

3.6 Complexity

A very important aspect of the GP techniques is the time complexity of the procedure used for computing the fitness of the newly created individuals.

The complexity of that procedure for the standard GP is $O(m * g)$, where m is the number of fitness cases and g is average number of nodes in the GP tree.

By contrast, the complexity of generating (by insertion or recombination) an individual in the liquid is only $O(m)$, because the liquid's item is generated by traversing an array of size m only once. The length of an item in the liquid is m .

Clearly, the use of the liquid could generate a small overhead of the LSGP when compared to the standard GP. Numerical experiments show (running times not presented due to the space limitation) that LSGP is faster than the standard GP, because the liquid part is considerably faster than the standard GP and many combinations performed in the liquid could lead to perfect solutions. However, it is very difficult to estimate how many generations we can run an LSGP program in order to achieve the same complexity as GP. This is why we restricted GP and LSGP to run the same number of generations.

4 Numerical Experiments

Several numerical experiments using LSGP are performed in this section by using the even-parity problem. The Boolean even-parity function of k Boolean arguments returns **T** (**True**) if an even number of its arguments are **T**. Otherwise the even-parity function returns **NIL** (**False**) [1].

4.1 Experimental Setup

General parameter of the LSGP algorithm are given in Table 2.

For the even-parity problems we use the set of functions (for both liquid and GP trees) $F = \{\text{AND, OR, NAND, NOR}\}$ as indicated in [1].

4.2 Summarized Results

Summarized results of applying Liquid State Genetic Programming for solving the considered problem are given in Table 3.

Table 2. General parameters of the LSGP algorithm

| Parameter | Value |
|---|---------------------------------------|
| Liquid's insertion probability | 0.05 |
| GP Selection | Binary Tournament |
| Terminal set for the liquid | Problem inputs |
| Liquid size | 2 * Number of inputs |
| Terminal set for the GP individuals | Liquid's items |
| Number of GP generations before updating the liquid | 5 |
| Maximum GP tree height | 12 |
| Number of runs | 100 (excepting for the even-8-parity) |

Table 3. Summarized results for solving the even-parity problem using LSGP and GP. Second column indicates the population size used for solving the problem. Third column indicates the number of generations required by the algorithm. The success rate for the GP algorithms is given in the fourth column. The success rate for the LSGP algorithms is given in the fifth column.

| Problem | Pop size | Number of GP generations | of GP rate (%) | success LSGP rate (%) | success |
|---------|----------|--------------------------|----------------|----------------------------|---------|
| even-3 | 100 | 50 | 42 | 93 | |
| even-4 | 1000 | 50 | 9 | 82 | |
| even-5 | 5000 | 50 | 7 | 66 | |
| even-6 | 5000 | 500 | 4 | 54 | |
| even-7 | 5000 | 1000 | - | 14 | |
| even-8 | 10000 | 2000 | - | 1 successful out of 8 runs | |

For assessing the performance of the LSGP algorithm in the case of even-parity problems we use the success rate metric (the number of successful runs over the total number of runs).

Table 3 shows that LSGP is able to solve the even-parity problems very well. Genetic Programming without Automatically Defined Functions was able to solve instances up to even-5 parity problem within a reasonable time frame and using a reasonable population. Only 8 runs have been performed for the even-8-parity due to the excessive running time. Note again that a perfect comparison between GP and LSGP cannot be made due to their different individual representation.

Table 3 also shows that the effort required for solving the problem increases with one order of magnitude for each instance.

5 Limitations of the Proposed Approach

The main disadvantage of the proposed approach is related to the history of the liquid which is not maintained. We cannot know the origin of an item from the

liquid unless we store it. We can also store the tree for each item in the liquid but this will lead to a considerable more memory usage and to a lower speed of the algorithm. Another possible way to have access to the liquid's history is to store the entire items ever generated within the liquid and, for each item, to store pointers to its parents. This is still not very efficient; however, it is better than storing the entire tree, because, in the second case, the subtree repetition is avoided.

6 Conclusions and Further Work

A new evolutionary technique called Liquid State Genetic Programming has been proposed in this paper. LSGP uses a special, dynamic memory for storing and manipulating the inputs.

LSGP has been used for solving several difficult problems. In the case of even-parity, the numerical experiments have shown that LSGP was able to evolve very fast a solution for up to even-8 parity problem. Note that the standard GP evolved (within a reasonable time frame) a solution for up to even-5 parity problem [1].

Further effort will be spent for improving the proposed Liquid State Genetic Programming technique. For instance, the speed of the liquid can be greatly improved by using Sub-machine Code GP [6], [7]. Many aspects of the proposed LSGP technique require further investigation: the size of the liquid, the frequency for updating the liquid, the operators used in conjunction with the liquid etc.

The proposed LSGP technique will be also used for solving other symbolic regression, classification [1] and dynamic (with inputs being variable in time) problems.

References

1. Koza, J. R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, MA, (1992)
2. Koza, J. R.: Genetic Programming II: Automatic Discovery of Reusable Subprograms, MIT Press, Cambridge, MA, (1994)
3. Maass, W. Natschlger, T., Markram, H.: Real-time computing without stable states: A new framework for neural computation based on perturbations, *Neural Computation*, Vol. 14 (2002) 2531-2560
4. Miller, J. F., Job, D., Vassilev, V. K.: Principles in the Evolutionary Design of Digital Circuits - Part I, *Genetic Programming and Evolvable Machines*, Vol. 1, Kluwer Academic Publishers (2000) 7-35
5. Natschlger, T., Maass, W., Markram H.: The "liquid computer": A novel strategy for real-time computing on time series. Special Issue on Foundations of Information Processing of *TELEMATIK*, Vol. 8, (2002) 39-43
6. Poli, R., Langdon, W. B.: Sub-machine Code Genetic Programming, in *Advances in Genetic Programming 3*, L. Spector, W. B. Langdon, U.-M. O'Reilly, P. J. Angeline, Eds. Cambridge:MA, MIT Press, chapter 13 (1999)

7. Poli, R., Page, J.: Solving High-Order Boolean Parity Problems with Smooth Uniform Crossover, Sub-Machine Code GP and Demes, *Journal of Genetic Programming and Evolvable Machines*, Kluwer (2000) 1-21
8. Prechelt, L.: PROBEN1: A Set of Neural Network Problems and Benchmarking Rules, Technical Report 21, (1994), University of Karlsruhe, (available from <ftp://ftp.cs.cmu.edu/afs/cs/project/connect/bench/contrib/prechelt/proben1.tar.gz>)
9. UCI Machine Learning Repository (available from www.ics.uci.edu/~mllearn/MLRepository.html)

Genetic Based Distribution Service Restoration with Minimum Average Energy Not Supplied

Thitipong Charuwat and Thanatchai Kulworawanichpong

Power and Energy Research Group, School of Electrical Engineering,
Institute of Engineering, Suranaree University of Technology,
Nakhon Ratchasima 30000, Thailand
Charuwat_t@hotmail.com, thanatch@sut.ac.th
<http://eng.sut.ac.th/ee>

Abstract. This paper presents optimal planning of tie-switch operation in an electric power distribution system under an emergency feed condition, i.e. operation during a post-fault condition. A heuristic fault isolation algorithm and a genetic-based service restoration algorithm are proposed and compared. With the proposed restoration algorithm, high reliable service of electric distribution systems is expected. To ensure a small number of customer interruption, average energy not supplied (AENS) is used as the objective function to be minimized. 25-node and 118-node distribution test feeders were employed for test. Satisfactory results show that the genetic approach is appropriate to a kind of tie-switch operation planning in order to minimize effects of a permanent fault on customer service interruption.

1 Introduction

Short-circuit or so-called “fault” conditions can occur unexpectedly in any part of an electric power distribution system at any time due to various physical failures. Such a case causes a fault current flowing through some power system equipment. The occurrence of the fault is harmful and must be isolated immediately by a set of protective devices. Over several decades, protective relaying has become the brain of power system protection [1]-[3]. Its basic function is to monitor abnormal operations as a “fault sensor” and the relay or other protective devices will open their contractors to separate a faulty part from the healthy parts of the network. To date, electric power distribution systems are bulky and complicated. These lead to the need for a large number of protective devices cooperating with one another to assure the secure and reliable operation of a whole. The research as presented in this paper is based on electric utility control practice for the applicability of a restoration algorithm. Since control operators can successfully deal with small-scale outages, such as feeder fault, a computer-based restoration algorithm should be primarily developed for large-scale outages in a large-scale distribution system to minimize customer service interruption [4]-[6]. Distribution feeder control is a real-time environment and an emergency under fault is a stressful situation. A computer-based restoration program can be helpful to system operators in which the proposing of various possible restoration scenarios

are available. The proposed restoration strategy used in many regional electric utilities nowadays is based on heuristic search guided by expert knowledge. In this paper, an intelligent optimization, genetic algorithms (GA), that guarantees the optimal solution has been proposed. Aiming to minimize customer service interruption, average energy not supplied (AENS) is employed to evaluate the fitness of each restoration scenario.

A heuristic fault isolation algorithm is introduced and described in the next section. Exploitation of GA to solve service restoration problems in order to minimize AENS is proposed in Sect. 3. To demonstrate this scheme, 25-node and 118-node IEEE standard test feeders were situated and the test results are presented in Sect. 4 while Sect. 5 gives conclusions.

2 Heuristic Fault Isolation Algorithm

When a faulty event was unexpectedly occurred at any feeder portion, system operators must decide which action to perform in order to keep operating the rest of the system under this emergency before it becomes haywire. To achieve this task a fault isolation algorithm is primarily required. Assume that a permanent short-circuit fault is occurred at a particular node and there is a fault location algorithm which was already performed successfully. The proposed heuristic fault isolation algorithm (HFIA) [7] applied herein is based on searching for the nearest switch in each connecting branch and then disconnecting them all. As a result, the faulty node is successfully isolated. This algorithm is similar to a spanning tree that expands from the faulty node through all connecting branches. It can be summarized step-by-step as follows.

1. Start from a faulty node k .
2. Check all connections between the faulty node and its adjacent nodes. Define Γ_k as a set of existing connections found.
3. Check through Γ_k , line-by-line: if there exists a switching device, disconnect it and remove it from Γ_k . Otherwise, go to step 4. When returned: if Γ_k is an empty set, go to step 5.
4. Sub-problem: recall step 2 but consider the node at the other end of the targeted line, says node i . To avoid confusion, define Φ_i instead of Γ_i as a set of existing connections found for this case. Repeat step 3.
5. Fault was isolated successfully.

As can be seen in Fig. 1, node 3 is the faulty node. For the first level of the spanning tree from node 3, Γ_3 is $\{d,e,h,k,q\}$. Checking through Γ_3 , S3 and S11 are the switching device of line d and e respectively. With operation of these two switches, branches d and e are disconnected and must be removed from Γ_3 . At this stage, Γ_3 is reduced to $\{h,k,q\}$. For line h, due to no switch protecting this line, Φ_h can be generated as $\Phi_h = \{i,j\}$. Switches S12 and S13 are found and therefore disconnected for branch i and j, respectively. Return to the main task. For line k, $\Phi_k = \{l,m\}$ and this leads to disconnection of switches S4 and S5. In the same manner, $\Phi_q = \{r,s,t\}$. Switches S8, S9 and S10 are operated. Eventually, Γ_3 is empty and the fault was completely isolated as shown in Fig. 1.

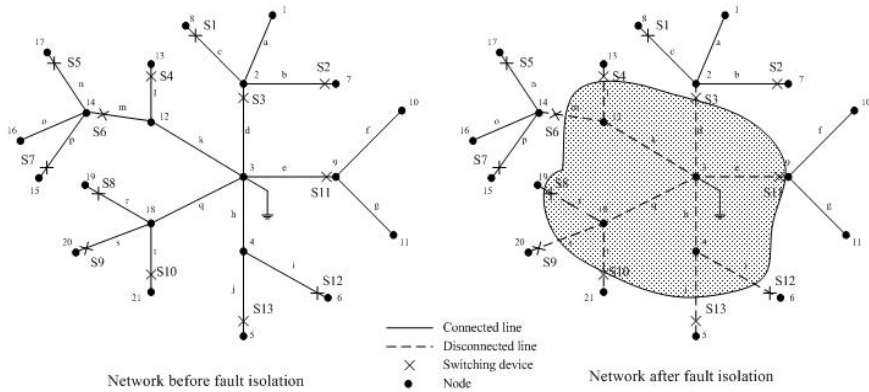


Fig. 1. Example of utilizing the heuristic fault isolation algorithm

3 Genetic Algorithms for Distribution Service Restoration

Service restoration problems [4]-[7] involve finding a set of switching status. Each bit of binary arrays represents a status of switches (switch opened or switch closed). To find a solution in order to minimize effects of fault isolation on customer interruption needs some efficient algorithm that can directly deal with binary variables. One of which widely used is genetic algorithms (GA) [8]-[9]. During the GA process, all variables must be coded into binary string. Each individual solution can be performed with genetic operators, e.g. crossover, mutation, etc, to create offspring. The offspring in binary code can be decoded into a real-valued variable in the end of the process in each generation. As can be seen, with a set of switching status each solution represents an already coded binary string that is ready to use in the GA process. Utilization of the GA for power restoration can reduce problem complexity in which the fault isolation and network reconfiguration to managing an emergency feed can be combined together. To formulate the restoration problem is obviously simple. All switches in the power distribution systems are set as bits to form a binary string. For example, assume that there are 4 switches (S1, S2, S3 and S4) in the system. A string representative consists of 4 bits. Each bit is either one (switch closed) or zero (switch opened). If S1 and S4 are in service while S2 and S3 are switched off, the string representative of this solution is 1001. Because the GA is very popular and widely used in most research areas where a intelligent search technique is applied, it can be summarized briefly as follows.

1. **Initialization:** Randomly initialize populations or chromosomes and set them as a search space and then evaluate their corresponding fitness value via the objective function.
2. **Evolution:** Apply the genetic operators to create an offspring population as the sequence below,
 - a. **Selection:** Form a set of mating pool with the same number of the population size by using a random procedure, e.g. the roulette-wheel

or tournament schemes [10], with the assumption that each chromosome has a different chance (probability to survive. The higher the fitness value, the higher the chance or probability.

- b. Crossover: This operation is applied to a subset of the mating pool by taking a pair of chromosomes called the parents. The parents will yield a pair of offspring chromosomes. This operation involves exchanging sub-string of the parent chromosomes. It is performed by choosing a random position in the string and then swapping either the left or right sub-strings of this position (one-point crossover) with its chromosome mate.
 - c. Mutation: For the chromosome to be mutated, the values of a few positions in the string are randomly modified. To prevent complete loss of the genetic information carried through the selection and crossover processes, mutation (if use at all) is limited to typically 2.5% of the population [10].
3. Fitness Test: Evaluate the fitness value for the generated offspring population.
 4. Convergence Check: Check for violation of all termination criteria. If not satisfied, repeat the evolution process.

An optimal solution is not unique. It depends on which criteria are made. In this paper, effects of fault on customer service interruption must be minimized. So a reliability index describing an amount of customers or loads is used to formulate the objective function. Average energy not supplied (AENS) [11]-[13] is defined by the following expression

$$\text{AENS} = \frac{\text{Total Energy Not Supplied}}{\text{Number of Customer Served}} \quad (1)$$

Given that a 9-bus power system consists of 7 load buses (at bus 2, 3, 4, 5, 7, 8 and 9) as show in Fig. 2. Assume that the line between bus 4 and 6 is disconnected for some reason. Loads at buses 7, 8 and 9 are not served with the total of 500 MW. Therefore, the AENS can calculated by the following expression

$$\text{AENS} = \frac{100 + 73 + 327}{7} = 71.4285$$

When the AENS is small, the average of interrupted loads in MWh per customer is also small. This implies that the smallest area of fault isolation could be achieved by minimizing the AENS. The objective function of the service restoration can be written as follows.

$$\begin{array}{ll} \text{Minimize} & \text{AENS} \\ \text{Subject to} & \text{Fault Isolation Criteria} \\ & \text{System Operation Limits} \end{array} \quad (2)$$

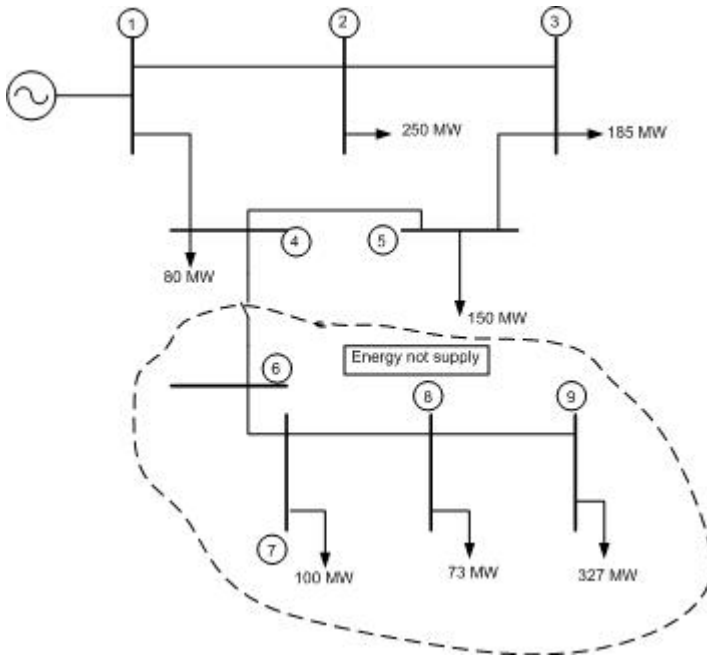


Fig. 2. Example of AENS calculation

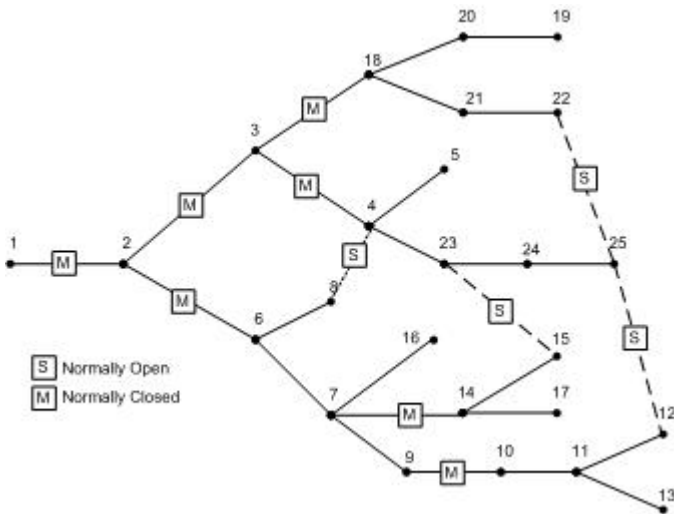


Fig. 3. 25-node IEEE standard test feeder [14]

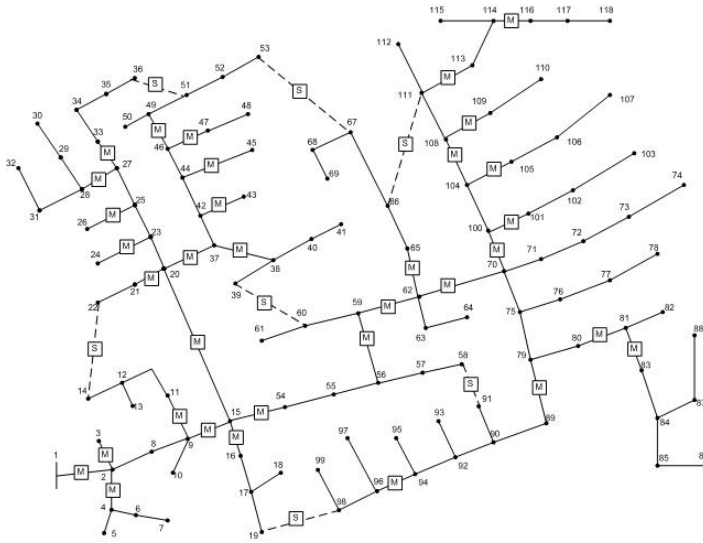


Fig. 4. 118-node IEEE standard test feeder [15]

4 Simulation Results

25-node and 118-node IEEE standard test feeders were situated for test as shown in Figs 3 and 4. In the 25-node test feeder, 11 switches are installed and 7 of them is connected to the system under normal operation. Whereas the 118-node test system consists of 42 switches. 36 switches are assigned to be switched on. In this work, parameter setting for the GA is as follows: {number of population = 50, crossover probability = 0.7, mutation probability = 0.005, maximum generation = 100} for each test case. The test case scenario is set by assigning fault at node 7 and node 55 for the 25-node and 118-node test systems, respectively. Table 1 shows results of minimum AENS using the GA for both cases.

Table 1. Optimal solution obtained by using the GA with minimum AENS

| Test system | AENS using HFIA | AENS using GA | % reduced AENS |
|-------------|-----------------|---------------|----------------|
| 25 nodes | 0.7201 | 0.15016 | 79.15 |
| 118 nodes | 0.8659 | 0.62712 | 27.51 |

The optimal solutions obtained by GA can be graphically presented in following single-line diagrams as shown in Figs 5 and 6 for 25-node and 118-node test cases, respectively. In addition, by using MATLAB™ Genetic Algorithms and Direct Search (GADS) TOOLBOX, convergences of the GA process is shown in Figs 7 and 8 for the IEEE 25-node and the IEEE 118-node test systems, respectively.

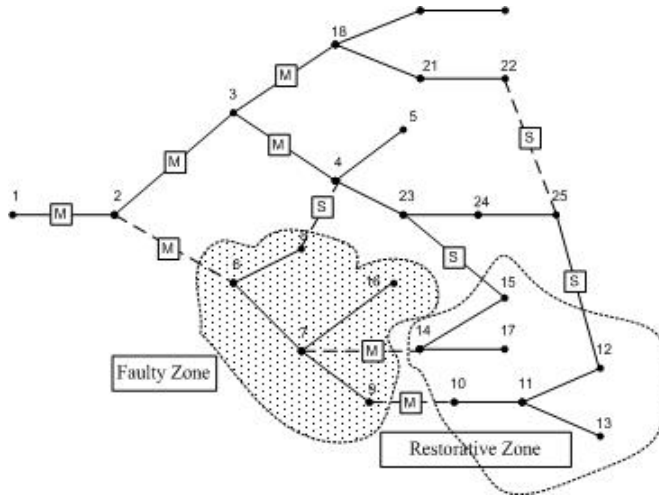


Fig. 5. Optimal solution of 25-node IEEE standard test feeder

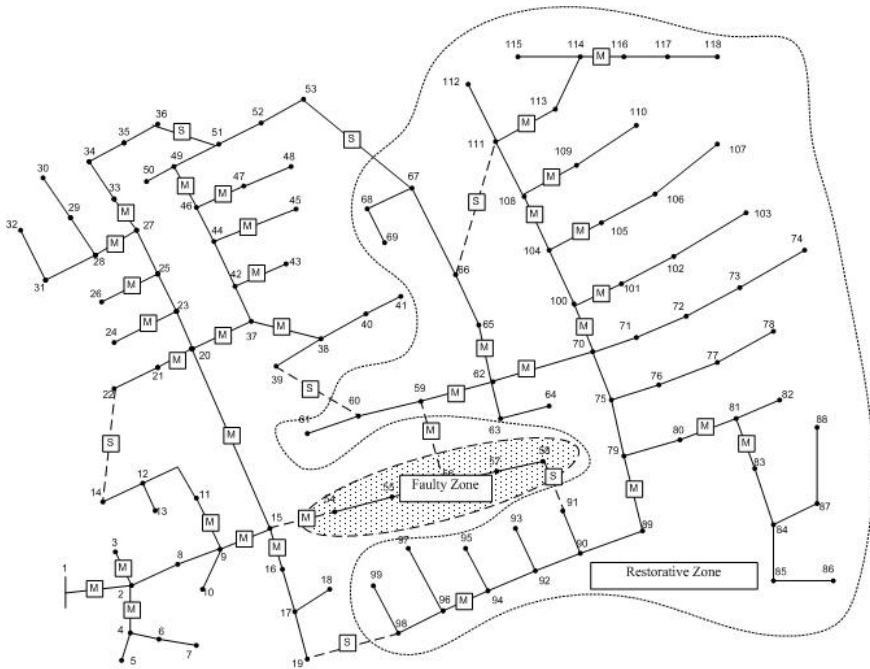


Fig. 6. Optimal solution of 118-node IEEE standard test feeder

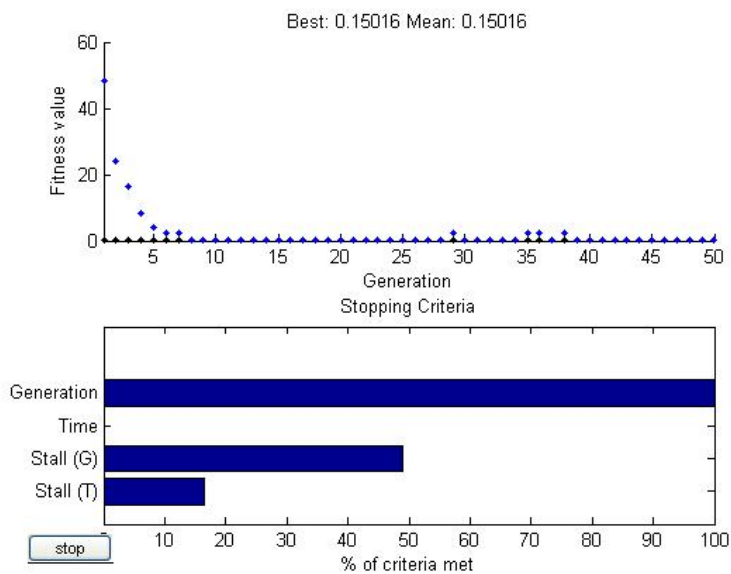


Fig. 7. Convergence of the GA optimizing process for the IEEE 25-node test system

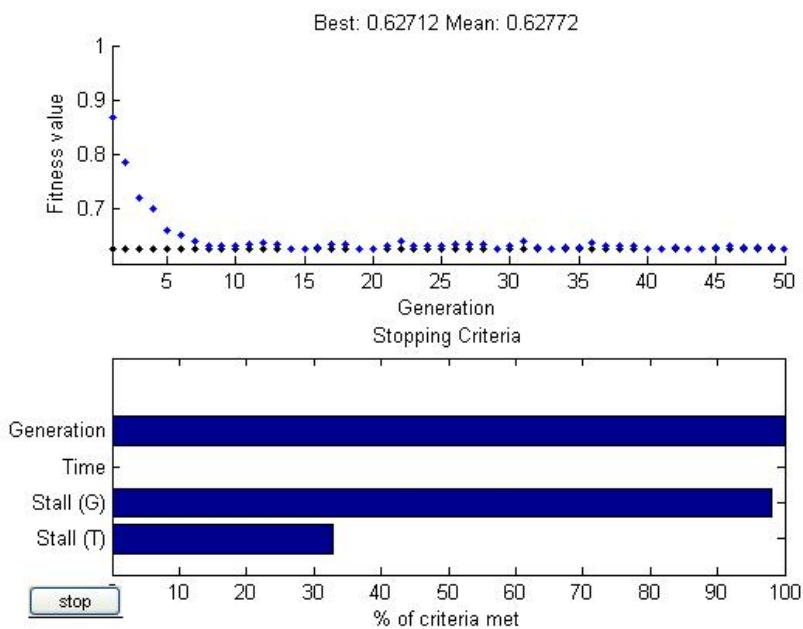


Fig. 8. Convergence of the GA optimizing process for the IEEE 118-node test system

Interestingly, before applying the GA to minimize AENS, all customers in faulty and restorative zones are completely blackout. After solving the optimization problem, the blackout area is confined as in the faulty zone only, where the restorative zone can serve their customers.

5 Conclusion

This paper presents optimal planning of tie-switch operation in an electric power distribution system under an emergency feed condition. A heuristic fault isolation algorithm (HFIA) can simply isolate the faulty node. However, it also results in a large area blackout. To reduce customer service interruption, a genetic-based service restoration algorithm is proposed. Due to the use of binary string to represent switching status, the GA is appropriate for this kind of problems. To ensure a small number of customer interruption, average energy not supplied (AENS) is used as the objective function to be minimized. 25-node and 118-node IEEE standard distribution test feeders were employed for test. As a result, the AENS after performing the intelligent optimization is reduced to approximately 40% of that obtained by the heuristic fault isolation algorithm (HFIA).

Acknowledgements. The authors would like to acknowledge the financial support from Suranaree University of Technology, Thailand.

References

1. Blackburn, J.L.: Protective Relaying. Marcel Dekker, 1987
2. Horowitz, S.H., Phadke, A.G.: Power System Relaying. Research Study Press, 1995
3. Qin, B.L., Guzman-Casillas, A., Schweitzer, E.O.: A New Method for Protection Zone Selection in Microprocessor-Based Bus Relays. *IEEE Trans on Power Delivery* 3 (2000), 876-887
4. Mun, K. J., Park, J.H., Kim, H., Seo, J.: Development of real-time-service restoration system for distribution automation system. *IEEE International Symposium on Industrial Electronics (ISIE 2001)*, June 2001, 1514 – 1519
5. Lehtonen, M., Matsinen, A., Antila, E., Kuru, J.: An advanced model for automatic fault management in distribution networks. *IEEE Power Engineering Society Winter Meeting (2000)*, January 2000, 1900 – 1904
6. Sarma, N.D.R., Prasad, V.C., Rao, P., Sankar, V.: A new network reconfiguration technique for service restoration in distribution networks. *IEEE Trans. on Power Delivery* 9 (1994), 1936 – 1942
7. Sudhakar, T. D., Vadivoo, N.S., Slochanal, S.M.R. : Heuristic based strategy for the restoration problem in electric power distribution system, *International Conference on Power System Technology (POWERCON 2004)*, November 2004, 635 - 639
8. So, C.W., Li, K.K., Lai, K.T., Fung, K.Y.: Application of Genetic Algorithm for Overcurrent Relay Coordination. *IEE Int. Conf. On Development in Power System Protection*, March 1997, 66-69

9. Choi, D., Kim, C., Hasegawa, J.: An application of genetic algorithms to the network re-configuration in distribution for loss minimization and load balancing problem. International Conference on Energy Management and Power Delivery (EMPD 1995), November 1995, pp. 376 – 381
10. Goldberg, D.E., Edward, D.: Genetic Algorithms in Search, Optimization and Machine Learning, Wiley. 1989
11. Li, W., Wang, P., Li, Z., Liu, Y.: Reliability evaluation of complex radial distribution system considering restoration sequence and network constraints. IEEE Trans. on Power Delivery 2 (2004), 753 – 758
12. He, Y., Anderson, G., Allan, R.N.: Modeling the impact of automation and control on the reliability of distribution systems. IEEE Power Engineering Society Summer Meeting, July 2000, 79 – 84
13. Billington, R., Allan, R.N.: Reliability Evaluation of Power Systems. Pitman Advanced Publishing. 1984
14. Goswami, S.K., Basu, S.K.: Direct solution of distribution systems. Proc. Inst. Electr. Eng. 138 (1991)
15. Distribution system analysis subcommittee: IEEE 118-node test feeder. IEEE Power Engineering Society

Multi-objective Feature Selection with NSGA II

Tarek M. Hamdani¹, Jin-Myung Won², Adel M. Alimi¹, and Fakhri Karray²

¹ REsearch Group on Intelligent Machines (REGIM)
University of Sfax, ENIS
BP. W-3038 – Sfax – Tunisia

{tarek.hamdani, adel.alimi}@ieee.org

² Pattern Analysis and Machine Intelligence Research Group,
Department of Electrical and Computer Engineering,
University of Waterloo, Waterloo, ON N2L 3G1, Canada
{jinmyung, karray}@pami.uwaterloo.ca

Abstract. This paper deals with the multi-objective definition of the feature selection problem for different pattern recognition domains. We use NSGA II the latest multi-objective algorithm developed for resolving problems of multi-objective aspects with more accuracy and a high convergence speed. We define the feature selection as a problem including two competing objectives and we try to find a set of optimal solutions so called Pareto-optimal solutions instead of a single optimal solution. The two competing objectives are the minimization of both the number of used features and the classification error using 1-NN classifier. We apply our method to five databases selected from the UCI repository and we report the results on these databases. We present the convergence of the NSGA II on different problems and discuss the behavior of NSGA II on these different contexts.

1 Introduction

The feature selection problem is defined as the selection of a subset of features of size d from an original set of size D with respect of $d \ll D$. This problem is very important in pattern recognition system especially when we are in presence of a very high number of features. With recent digitalization technologies and analysis, the number of presented features can be of a very high number and we can frequently have redundant or ambiguous information. This is why number of approaches was developed to deal with features selection problem [1].

Evolutionary techniques were intensively used for feature selection to solve the combinatory problem and to provide efficient exploration of the solutions' space and to provide one optimal solution with the maximum classification performance [2]. Genetic Algorithms (GAs) were specifically developed for this task and results provided by GA solution were more efficient than classical methods developed for feature selection as confirmed in [3], [4].

Multi-objective approach, which was intensively studied in the recent years [5], [6], was applied in many fields in relation with the pattern recognition problem. In [7] authors used the multi-objective approach in the field of features selection for creating

classifiers' ensembles and it was developed and applied to handwriting words recognition in [8]. The problem of nuclear transients' classification was also addressed in [9] using GA based feature selection. GA based approaches was also use in [10] with a clustering context for pattern recognition in the case of an environmental problem.

Naturally, features selection problem can be defined as a multi-objective problem dealing with two competing objectives. Consequently, an optimal feature set has to be of a minimal number of features and have to produce the minimum classification error. Number of works was developed to provide a multi-objective solution for feature selection problem as in [11]. In [12] authors applied mutli-objective approach for features selection to recognize digits and they defined for this their method that was based on the NSGA algorithm of [13]. We can also refer the work of [14] in which authors propose a hierarchical multi-objective solution for feature selection.

In this paper we resolve the features selection problem with the use of NSGA II [15] the latest developed algorithm for multi-objective problems. We define the feature selection problem as a task of two competing objectives and we try to minimize both the number of used features and the classification error to find pareto-optimal solutions for five pattern recognition problems. The results found on the tested databases were very interesting. In Sect. 2, we describe the proposed definition of the features selection problem with NSGA II. In Sect. 3, we explain how we establish our experimental process and we report the experimental results for different pattern recognition problems. Sect. 4 draws conclusions and suggests several future works.

2 Feature Selection Problem with NSGA II

We implemented the feature selection problem referring to a simple coding scheme and we used binary chromosomes to present even the feature was selected or not.

Two competing objectives were defined; the first was the minimization of the number of used features; the second was the classification error. The classification error was computed on each database using the 1-NN classifier to evaluate the discrimination value of the each selected set of features.

We used 1-NN as classifier to evaluate the performance of a given feature set identified by a chromosome on different training and testing sets. The 1-NN classifier returns a real variable belonging to $[0,1]$ and denoting the classification error on a testing data set.

Furthermore, to decrease the computational complexity of the pattern recognition problem we have implemented a special classifier 1-NN that saves the computational time by computing just the necessary distances pointed by the selected feature. This improvement was implemented by means of the integration of a data structure completing the chromosome encoding containing the indexes of the selected sequence of features. This saves the computational time of the whole classification process.

The NSGA II [15] was recently implemented by Deb to improve the first version of the algorithm. Deb has show that the new algorithm outperforms the first version of NSGA and has a lower computational complexity. The principal of this algorithm is to use the fast non-dominated sorting technique and a crowding distance to construct

the population's fronts that dominate each other in a non-dominating rank. The algorithm uses a niche technique and a speciation technique to preserve diversity and to find the best population. One of the most important proposition of the NSGA II is that it's proposes to modify the non-dominating sorting process to accelerate it by the definition of the fast non-dominating sorting that decreases the complexity of the algorithm from $O(MN^3)$ to $O(MN^2)$.

We implemented this algorithm and we tested its efficiency on different contexts and we report the results in the next section. The used crossover technique was the uniform crossover consisting on replacing genetic material of the two selected parents uniformly in several points. The mutation operator used in this work was implemented as conventional mutation operator operating on each bit separately and changing randomly its value.

3 Experimental Results

In the following subsections we will describe the experimental setting with a description of the used databases and we present the results found in different contexts defined by the pattern recognition problems.

3.1 Experimental Setting

For the experimentations we used five datasets selected from the UCI repository [16] and we implemented NSGA II with the flowing parameters:

- Population size: 50
- Crossover probability: 0.5
- Mutation probability: $1/\text{number of features}$

Details of the used databases are presented in the following table (see Table 1) with the reference recognition rate computed for each pattern recognition problem using all the predisposed features.

Table 1. Detailed Information about Used Databases

| Database name | Training points | Testing points | Number of features | Number of classes | Recognition rate |
|------------------|-----------------|----------------|--------------------|-------------------|------------------|
| <i>Segment</i> | 700 | 700 | 19 numerical | 7 | 91,00 |
| <i>Satellite</i> | 600 | 600 | 36 numerical | 6 | 81,67 |
| <i>Letter</i> | 2 600 | 2 600 | 16 numerical | 26 | 86,04 |
| <i>MNIST</i> | 1 000 | 1 000 | 114 numerical | 10 | 81,00 |
| <i>DNA</i> | 300 | 300 | 180 logical | 3 | 62,67 |

3.2 NSGA II Results

Five databases were used to perform the experimentation on the NSGA II for pattern recognition and results observed by the pareto-optimal population are given in respecting three categories; the first category presents the pareto-optimal solutions for the simple structured problems (Segment and Satellite); the second category presents

results of the Letter database that can be considered as a complex pattern recognition problem; the third deals with the results of NSGA II for the MNIST and DNA databases. These problems have the specificity to be of a high number of features and are defined for a high number of data points.

For all the experimentations we will present two graphs with the set of pareto-optimal solutions in first iteration and after convergence of NSGA II. We will also present with the pareto-optimal solutions in Fig. 2 a reference diamond point which will represent the original solution defined by all the features and its associated error rate to compare it to the given solutions.

The x axis will present the number of used features and the y axis will measure the error rate associated to each defined solution. In the following graphs we will not present all the population given by the NSGA II and we will just use the first front population composed of individuals that dominate all the other suboptimal solutions.

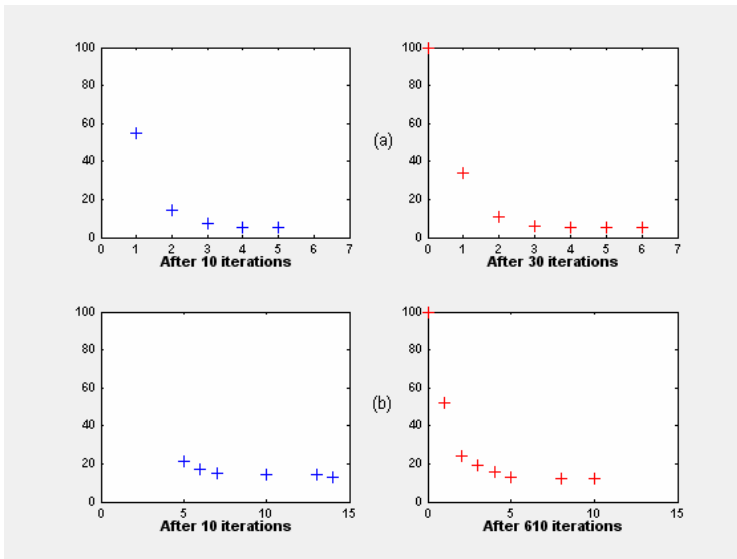


Fig. 1. The graphic presents the results for small databases using NSGA II. Results in (a) are reported for the Segment database. In (b) we reported results for the Satellite database. The first graph for each database is associated to 10 first iterations of the algorithm. The second graph presents the pareto-optimal subset after convergence.

In these results we can observe that the NSGA II converges from the first iteration and the pareto-optimal solutions were progressively ameliorated. Solutions given after convergence are more diversified and give the used more possibilities to choose his suboptimal solution. The earlier convergence speed can be explained by the simplicity of the used databases.

The next experimentation will concerns a very specific case in which the data are of a high number but we manipulate here a limited number of features results presented in Fig. 2.

This experimentation is specific due to the high number of classes and the high number of used data, which was relatively high and increases the complexity of the pattern recognition problem. NSGA II has reduced this complexity proposing a set of high quality solutions. In spite of the high complexity of the pattern recognition problem the NSGA II has converge from the first iterations to a relatively good set of solutions and the convergence of the algorithm was also relatively early.

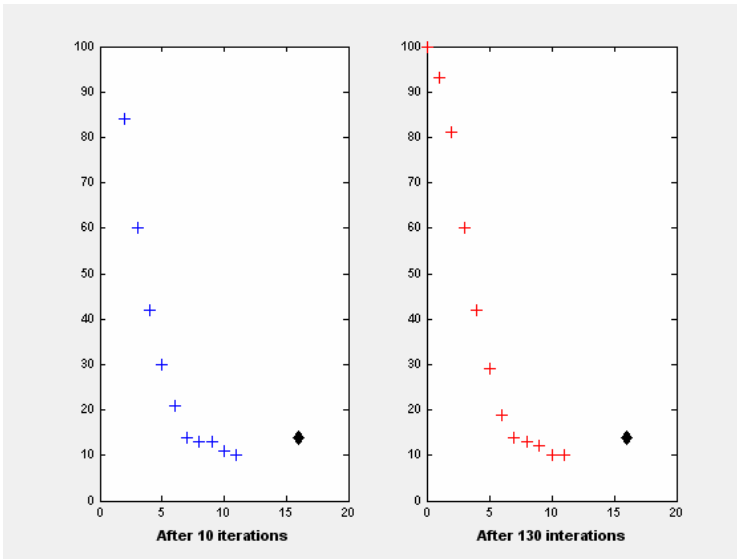


Fig. 2. Graphic presents results of NSGA II on the Letter database. The algorithm converges from the first iterations and it tries to ameliorate and increase the number of pareto-optimal solutions.

The NSGA II algorithm begins with proposing a set of non optimal solutions and will improve it and increase the number of solutions to be more efficient (see Fig.3). In these two cases the convergence of NSGA II wasn't evident from the first iterations.

MNIST and DNA databases are considered as complex databases for two points of view; the first is the high number of data points and the second is the abundance of defined features. In such kind of problems NSGA II gives amelioration and spends more time to converge. We can see in Fig. 3 that after convergence solutions are more distributed in the space and are much more optimal that in first iterations.

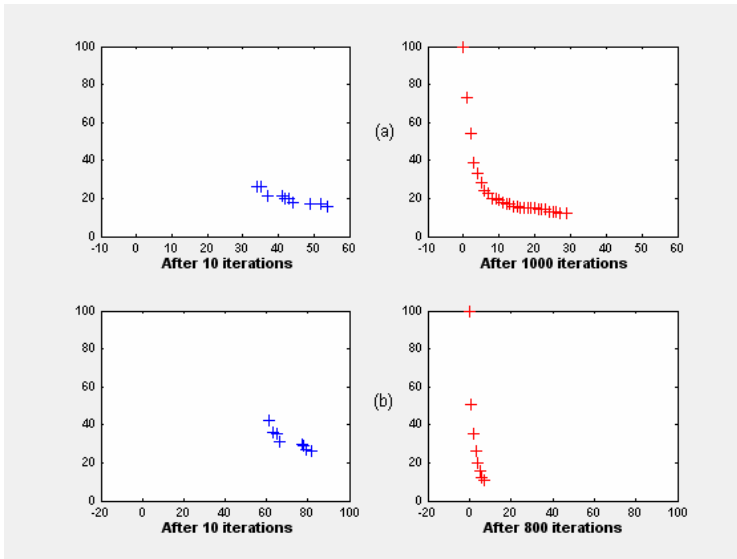


Fig. 3. Results of the NSGA II on MNIST and DNA databases are presented on the graph and we can clearly see the improvement of the pareto-optimal set

3.3 Convergence and Computational Time

The computational time of the presented solution is reported in the Table 2.

Table 2. Computational time for the NSGA II in the different conducted experimentations

| Database | Number of iterations for convergence | Computational time for 10 iterations in minutes |
|------------------|--------------------------------------|---|
| <i>Segment</i> | 30 | 1 |
| <i>Satellite</i> | 610 | 1 |
| <i>Letter</i> | 130 | 7 |
| <i>MNIST</i> | 1000 | 5 |
| <i>DNA</i> | 800 | < 1 |

From this table we can see that the computational time of the NSGA II depends on the complexity of the defined pattern recognition problem. The computational time was higher than the other databases for complex databases as Letter and MNIST.

Convergence of the NSGA II can be measured referring to the necessary number of iterations for the algorithm to find the pareto-optimal set. According to this criterion, the convergence can be directly related to the number of defined features. This can be considered with the complexity and the eventual correlations existing between features.

4 Conclusion

In this paper we proposed the use of NSGA II the latest developed algorithm for features selection. The proposed solution was tested on five standard pattern recognition problems selected from the UCI repository and we presented the quality of the pareto-optimal sets given by NSGA II. We analyzed the quality of the given solution on three different types of databases and we found that the NSGA II provides really very good solutions from the first iterations for the simple datasets and then it continuously ameliorates the quality of the optimal set. For difficult datasets we observe a quick convergence on the datasets having relatively small number of features. For the third type of problems, which is characterized by a high number of features, the convergence time in term of number of iteration is more important and the improvement is more enhanced. Future works on these results will be conducted to accelerate more the convergence of the NSGA II for the specific case of discrete variables.

Acknowledgments. The authors would like to acknowledge the financial support of this work by grants from the General Direction of Scientific Research and Technological Renovation (DGRSRT), Tunisia, under the ARUB program 01/UR/11/02.

References

1. Dash M. and Liu H., "Feature Selection for Classification," *Intelligent Data Analysis* 1, pp. 131–156, 1997.
2. Kim G., and Kim S., "Feature Selection Using Genetic Algorithms for Handwritten character Recognition," 7th Int'l Workshop on Frontiers in Handwriting Recognition, pp. 103–112, Amsterdam, 2000.
3. Oh, I-S., Lee, J-S., and Moon, B-R., "Hybrid Genetic Algorithms for Feature Selection," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 11, November 2004.
4. Hamdani M.T., Alimi M.A., and Karray F., "Distributed Genetic Algorithm with Bi-Coded Chromosomes and a New Evaluation Function for Features Selection," *Proc. IEEE Congress on Evolutionary Computation*, pp. 2596–2603, 2006.
5. Cvetkovic, D., Parmee, I.C., "Preferences and their application in evolutionary multiobjective optimization," *Evolutionary Computation, IEEE Transactions on*, Volume 6, Issue 1, pp. 42-57, Feb. 2002.
6. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G., "Performance assessment of multiobjective optimizers: an analysis and review," *Evolutionary Computation, IEEE Transactions on*, Vol 7, Issue 2, pp: 117 - 132, April 2003.
7. Morita, M., Oliveira, L.S., Sabourin, R., "Unsupervised feature selection for ensemble of classifiers," *Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth International Workshop on*, pp. 81-86, 26-29 Oct. 2004.
8. Morita, M., Sabourin, R., Bortolozzi, F., Suen, C.Y., "Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition," *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on*, pp: 666-670, 3-6 Aug. 2003.

9. Zio, E., Baraldi, P., Pedroni, N., "Selecting features nuclear transients classification by means of genetic algorithms," Nuclear Science, IEEE Transactions on, Vol 53, Issue 3, Part 3, pp: 1479-1493, June 2006.
10. Devogelaere, D., Rijckaert, M., "Scalars, a way to improve the multi-objective prediction of the GAdC-method," Neural Networks, 2000. Proceedings. Sixth Brazilian Symposium on, pp. 56-60, 22-25 Nov. 2000.
11. Lac, H.C., Stacey, D.A., "Feature subset selection via multi-objective genetic algorithm," Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on, Vol 3, pp. 1349-1354, 31 July - 4 Aug. 2005.
12. Oliveira, L.S.; Sabourin, R.; Bortolozzi, F.; Suen, C.Y., "Feature selection using multi-objective genetic algorithms for handwritten digit recognition," Pattern Recognition, 2002. Proceedings. 16th International Conference on. Vol 1, pp: 568-571, 11-15 Aug. 2002.
13. Srinivas, N., and Deb, K., "Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms," Journal of Evolutionary Computation, Vol. 2, No. 3, pp. 221-248, 1998.
14. Oliveira, L.S., Sabourin, R., Bortolozzi, F., Suen, C.Y., "Feature selection for ensembles: a hierarchical multi-objective genetic algorithm approach," Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on, pp. 676-680, 3-6 Aug. 2003.
15. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., "A fast and elitist multiobjective genetic algorithm: NSGA-II," Evolutionary Computation, IEEE Transactions on, Volume 6, Issue 2, pp. 182 - 197, April 2002.
16. Newman D.J., Hettich S., Blake C.L., and Merz C.J., "UCI Repository of machine learning databases" [<http://www.ics.uci.edu/~mllearn/MLRepository.html>]. Irvine, CA: University of California, Department of Information and Computer Science, 1998.

Design of 2-D IIR Filters Using Two Error Criteria with Genetic Algorithm

Felicja Wysocka-Schillak

University of Technology and Life Sciences,
Institute of Telecommunications,
al. Prof. S. Kaliskiego 7, 85-796 Bydgoszcz, Poland
felicja@mail.atr.bydgoszcz.pl

Abstract. The paper presents a method for designing 2-D IIR filters with a quadrantly symmetric magnitude response. The method is based on two error criteria, i.e., equiripple error criterion in the passband and least-squared error criterion in the stopband. Two objective functions are introduced and the filter design problem is transformed into an equivalent bicriterion optimization problem. The stability of the filter is ensured by explicitly including stability constraints in the considered optimization problem. A two-step solution procedure of the considered problem is proposed. In the first step, a genetic algorithm is applied. The final point from the genetic algorithm is used as the starting point for a local optimization method. Two design examples are given to illustrate the proposed technique. A comparison with a 2-D IIR filter designed using LS approach is also presented.

1 Introduction

Two dimensional (2-D) digital filtering is one of the most important processing techniques in 2-D digital signal processing. 2-D filters have many important applications, e.g., in image processing and in seismic, radar and sonar signal processing.

Digital filters can be classified into two groups, i.e., finite impulse response (FIR) filters and infinite impulse response (IIR) filters. FIR filters can have exactly linear-phase response and they are free from stability problems [7]. In comparison with FIR filters, IIR filters offer better selectivity and improved computational efficiency due to the significant reduction in the number of multipliers [5,7,14]. However, the major problem encountered in the design of IIR filters is stability. IIR filters are useful in wide range of applications where high selectivity and efficient processing of discrete signals are desired [8].

Techniques for designing 2-D IIR digital filters have been developed extensively for several years [2,4,5,6,7,8,9,10,11,13,14]. The design of 2-D IIR filters is more complicated than the design of 2-D FIR filters. The transfer functions of IIR filters are rational functions and the resulting optimization problems are

highly nonlinear and computationally intensive. Besides, IIR filters can be unstable and that is why the stability constraints must be incorporated into IIR filter design problems.

2-D IIR filters have usually been designed using either the minimax or least-squared (LS) error criteria [7]. In case of the minimax design, the maximum error is smaller than in case of the LS design. However, the minimax design is complicated and computationally intensive. In many cases, the design based on the equiripple error criterion in the passband and LS criterion in the stopband is much more appropriate than the pure minimax or LS design. A detailed discussion on this problem can be found in [1].

In the paper, a new approach for the design of 2-D IIR filters with a quadrantly symmetric magnitude response is presented. This approach is based on the equiripple error criterion in the passband and the LS criterion in the stopband. In this approach, the approximation problem is transformed into an equivalent bi-criterion optimization problem that is converted into a single criterion one using the weighted sum strategy. The obtained objective function is then minimized subject to several constraints. These constraints include stability constraints and a constraint on the maximum allowable approximation error in the passband. The solution of the considered problem is achieved using a two-step procedure. In the first step, a genetic algorithm (GA) [3] is applied. GAs are probabilistic search techniques based on the mechanics of natural genetics and natural selection. They have strong robustness and general utility. Therefore, GAs are often used for solving difficult nonlinear optimization problems and multi-objective optimizations. The final point from the GA is used as the starting point for a local optimization method. Using a local optimization method, when the solution is close to the optimum, results in improving the speed of convergence. Two numerical examples are presented to illustrate the proposed technique. The results are compared with those obtained using the LS approach.

2 Formulation of the Design Problem

Let $H(z_1, z_2)$ be the transfer function of a 2-D IIR filter. Assume $H(z_1, z_2)$ has a separable denominator. This assumption imposes a constraint on the type of the IIR filter, because only IIR filters with quadrantly symmetric frequency response have the separable denominator [2,8,13]. The class of quadrantly symmetric 2-D IIR filters covers practically all types of 2-D IIR filters that have been found useful in 2-D signal processing applications [8].

A two-variable function $F(\omega_1, \omega_2)$ possesses quadrantal symmetry if it satisfies the following condition [6]:

$$F(\omega_1, \omega_2) = F(-\omega_1, \omega_2) = F(\omega_1, -\omega_2) = F(-\omega_1, -\omega_2) \quad (1)$$

If $F(\omega_1, \omega_2)$, in addition, satisfies

$$F(\omega_1, \omega_2) = F(\omega_2, \omega_1) \quad (2)$$

then it has octagonal symmetry.

The transfer function of a quadrantly symmetric 2-D IIR filter can be expressed as

$$H(z_1, z_2) = \frac{B(z_1, z_2)}{D(z_1)D(z_2)} \tag{3}$$

where

$$B(z_1, z_2) = \sum_{i=0}^n \sum_{k=0}^n b_{ik} z_1^{-i} z_2^{-k} \tag{4}$$

and

$$D(z_j) = \sum_{i=0}^m g_i z_j^{-i}, \quad j = 1, 2 \tag{5}$$

with $g_0 = 1$.

In case of an octagonally symmetric 2-D IIR filter, the transfer function can be written in the form [13]:

$$\begin{aligned} H(z_1, z_2) &= \frac{Q(z_1 + z_1^{-1}, z_2)Q(z_2 + z_2^{-1}, z_1)}{D(z_1)D(z_2)} = \\ &= \frac{(\sum_{i=0}^M \sum_{j=0}^N a_{ij} (z_1 + z_1^{-1})^i z_2^j)(\sum_{i=0}^M \sum_{j=0}^N a_{ij} (z_2 + z_2^{-1})^i z_1^j)}{(z_1^K + \sum_{i=0}^{K-1} d_i z_1^i)(z_2^K + \sum_{i=0}^{K-1} d_i z_2^i)} \end{aligned} \tag{6}$$

An octagonally symmetric 2-D IIR filter design problem is to determine the coefficients a_{ij} and d_i of the stable transfer function $H(z_1, z_2)$ such that the resulting frequency response is the best approximation of the desired frequency response in the given sense.

The advantage of using the transfer function given by (6) is that the number of independent coefficients to optimize is significantly reduced in comparison with (3).

The denominator polynomial $D(z_1, z_2) = D(z_1)D(z_2)$ has to satisfy the conditions for the stability. $D(z_1, z_2)$ is stable if and only if both $D(z_1)$ and $D(z_2)$ are stable [9][13]. A 1-D polynomial $D(z)$ is stable if its zeros are in the region $\{z : |z| < 1\}$ [7].

Let $\mathbf{Y} = [a_{00}, a_{01}, \dots, a_{NM}, d_0, d_1, \dots, d_{K-1}]^T$ be a vector of the transfer function coefficients.

Assume that the continuous (ω_1, ω_2) - plane is discretized by using a $K_1 \times K_2$ rectangular grid $(\omega_{1k}, \omega_{2l}), k = 0, 1, \dots, K_1 - 1, l = 0, 1, \dots, K_2 - 1$.

The desired magnitude response $A_d(\omega_{1k}, \omega_{2l})$ of the 2-D filter is:

$$A_d(\omega_{1k}, \omega_{2l}) = \begin{cases} 1 & \text{for } (\omega_{1k}, \omega_{2l}) \text{ in the passband } P, \\ 0 & \text{for } (\omega_{1k}, \omega_{2l}) \text{ in the stopband } S. \end{cases} \tag{7}$$

Let $A(\omega_{1k}, \omega_{2l}, \mathbf{Y}) = |H(e^{j\omega_1}, e^{j\omega_2}, \mathbf{Y})|$ denote the amplitude response of the filter obtained using the coefficients given by vector \mathbf{Y} .

In the passband P , the approximation is to be equiripple, so the error function $E(\omega_{1k}, \omega_{2l}, \mathbf{Y})$ is:

$$E(\omega_{1k}, \omega_{2l}, \mathbf{Y}) = A(\omega_{1k}, \omega_{2l}, \mathbf{Y}) - A_d(\omega_{1k}, \omega_{2l}), \quad \omega_1, \omega_2 \in P. \tag{8}$$

In the stopband S , the LS error $E_2(\mathbf{Y})$ to be minimized is:

$$E_2(\mathbf{Y}) = \sum_{(\omega_{1k}, \omega_{2l}) \in S} [A(\omega_{1k}, \omega_{2l}, \mathbf{Y}) - A_d(\omega_{1k}, \omega_{2l})]^2 \tag{9}$$

The filter design problem can be formulated as follows: For desired amplitude response $A_d(\omega_{1k}, \omega_{2l})$ defined on a rectangular grid $K_1 \times K_2$, and and given degrees of numerator and denominator find a vector \mathbf{Y} for which the considered filter is stable and the error function $E(\omega_{1k}, \omega_{2l}, \mathbf{Y})$ is equiripple in the passband and, simultaneously, the LS error $E_2(\mathbf{Y})$ is minimized in the stopband. Optionally, the following condition on the maximum allowable approximation error $\delta > 0$ in the passband can be additionally imposed:

$$\forall \omega_{1k}, \omega_{2l} \in P \quad |A(\omega_{1k}, \omega_{2l}, \mathbf{Y}) - A_d(\omega_{1k}, \omega_{2l})| \leq \delta \tag{10}$$

Adding the above condition results in obtaining the magnitude ripple equal or smaller than δ .

3 Transformation of the Problem

In order to solve the considered design problem, we transform it into an equivalent bicriterion optimization problem. We introduce two objective functions $X_1(\mathbf{Y})$ and $X_2(\mathbf{Y})$. We assume that the function $X_1(\mathbf{Y})$ has the minimum equal to zero when the error function $E(\omega_{1k}, \omega_{2l}, \mathbf{Y})$ is equiripple in the passband. The error function $E(\omega_{1k}, \omega_{2l}, \mathbf{Y})$ is equiripple in the passband when the absolute values $\Delta E_i(\mathbf{Y})$, $i = 1, 2, \dots, J$, of all the local extrema of the function $E(\omega_{1k}, \omega_{2l}, \mathbf{Y})$ in the passband, as well as the maximum value $\Delta E_{J+1}(\mathbf{Y})$ of $E(\omega_{1k}, \omega_{2l}, \mathbf{Y})$ at the passband edge are equal, i.e.:

$$\Delta E_i(\mathbf{Y}) = \Delta E_k(\mathbf{Y}), \quad k, i = 1, 2, \dots, J + 1. \tag{11}$$

In order to find a vector \mathbf{Y} , for which the conditions (III) hold, we introduce an objective function $X_1(\Delta E_1, \Delta E_2, \dots, \Delta E_{J+1})$ defined as follows:

$$X_1(\Delta E_1, \Delta E_2, \dots, \Delta E_{J+1}) = \sum_{i=1}^{J+1} (\Delta E_i - R)^2, \tag{12}$$

where:

$$R = \frac{1}{J+1} \sum_{k=1}^{J+1} \Delta E_k. \tag{13}$$

is the arithmetic mean of all ΔE_k , $k = 1, 2, \dots, J + 1$.

The function X_1 is non-negative function of $\Delta E_1, \Delta E_2, \dots, \Delta E_{J+1}$ and it is equal to zero if and only if $\Delta E_1 = \Delta E_2 = \dots = \Delta E_{J+1}$. As $\Delta E_1, \Delta E_2, \dots, \Delta E_{J+1}$ are the functions of the vector \mathbf{Y} , the function X_1 can be used as the first objective function in our bicriterion optimization problem. As the second objective function we apply the LS error $E_2(\mathbf{Y})$, so $X_2(\mathbf{Y}) = E_2(\mathbf{Y})$.

We convert the bicriterion optimization problem into a single criterion one using the weighted sum strategy. The equivalent optimization problem can be stated as follows: For given filter specifications and a weighting coefficient β find a vector \mathbf{Y} such that the function

$$X(\mathbf{Y}, \beta) = \beta X_1(\mathbf{Y}) + X_2(\mathbf{Y}) \quad (14)$$

is minimized, when the following stability constraints are given:

$$D(z_1) \neq 0 \quad \text{for } |z_1| \geq 1 \quad (15)$$

$$D(z_2) \neq 0 \quad \text{for } |z_2| \geq 1 \quad (16)$$

4 Two-Step Solution Procedure

In the optimization problem formulated in the previous section, the objective function is highly nonlinear, may have many local minima and has high dimensionality. Besides, the stability constraints on the filter coefficients are imposed. Local optimization methods may succeed in certain cases, but they generally are less suited for solving such difficult optimization problems. Global methods, such as GAs, are particularly effective when the goal is to find an approximate global minimum in case of high-dimensional, difficult optimization problems, objective functions that can have many local minimas and multi-objective optimizations. GAs are also largely independent of the initial conditions. That is why GAs are well suited for solving the considered optimization problem.

GAs are stochastic search and optimization techniques based on the mechanism of natural selection where stronger individuals would likely be the winners in a competing environment. A simple GA relies on the processes of reproduction, crossover and mutation to reach the global or "near-global" minimum. GAs operate on fixed length strings (chromosomes) representing possible solutions of a given optimization problem. To start implementing a GA, an initial population is considered. Successive generations are produced by manipulating the solutions in the current populations. Each solutions has a fitness (an objective function) that measures its competence. New solutions are formed using crossover and mutation operations. According to the fitness value, a new generation is formed by selecting the better chromosomes from the parents and offspring, and rejecting other so as to keep the population size constant. The algorithm converges to the best chromosome, which represents the solution of the considered optimization problem. The detailed description of a simple GA can be found in [3].

In order to solve the optimization problem formulated in the previous section, a two-step procedure is proposed, i.e., a hybridization of the GA and a local optimization method. Such hybridization is described by Golberg [3]. It is useful in our case because GAs are slow in convergence, especially when the solution is close to the optimum. In order to improve the speed of convergence, after a specified number of generations in the GA has been reached, a local optimization method, i.e., the Davidon, Fletcher, and Powell (DFP) method is applied to solve

the considered problem. The final point from the GA is used as the starting point for the DFP method. The DFP method is a quasi-Newton method which approximates the inverse Hessian matrix [12].

Numerical calculations have shown that it is possible to achieve better convergence if, instead of the minimization problem formulated in the previous section, we apply the GA to the following least square approximation problem

$$E_2(\mathbf{Y}) = \sum_{(\omega_{1k}, \omega_{2l}) \in P \cup S} [H(\omega_{1k}, \omega_{2l}, \mathbf{Y}) - H_d(\omega_{1k}, \omega_{2l})]^2 \quad (17)$$

Then, the solution of this problem is used as a starting point for solving the problem of minimizing $X(\mathbf{Y})$ using the DFP method. The local extrema of the error function $E(\omega_{1k}, \omega_{2l}, \mathbf{Y})$ are determined by searching the grid.

In applying the GA, the choice of the probability of crossover, the probability of mutation as well as the choice of the population size are very important. Their settings are dependent on the form of objective function. In the developed program, the population size is 30, the probability of crossover is 0.8, and the probability of mutation is 0.01.

The constrained optimization problem has been transformed into an unconstrained problem using penalty function technique.

5 Design Examples

In this section, two design examples are presented to illustrate the performance of the proposed approach. In both examples, a square grid of 101×101 points is used for discretizing the (ω_1, ω_2) - plane. The desired magnitude response is 1 in the passband P , 0 in the stopband S and varies linearly in the transition band Tr . The weighting coefficient is $\beta = 2 \times 10^4$.

As the first example, we design a diamond-shaped, lowpass 2-D IIR filter with the passband situated between the points $(0, 0.35\pi)$, $(0.35\pi, 0)$, $(0, -0.35\pi)$ and $(-0.35\pi, 0)$ on the $(\omega_{1k}, \omega_{2l})$ -plane. The width of the transition band is 0.14π . The filter is designed with $M = 1$, $N = 9$, $K = 7$, and $\delta = 0.07$. The magnitude response of the resulting filter is shown in Fig. 1. The designed filter is stable. The maximum pole magnitude is 0.778.

In order to compare the resulting filter with the filter obtained using the LS approach, the LS filter was designed for the same filter specifications. In case of the proposed approach, the maximum approximation error in the passband is $\delta = 0.07$. For the LS filter, $\delta = 0.120$. Note that in case of the proposed approach, the maximum approximation error δ is considerably smaller than in case of the LS approach.

As the second example, we design a circularly symmetric, lowpass 2-D IIR filter. The passband of the filter is a circular region centered at $(0, 0)$ with a radius $r_p = 0.5\pi$. The stopband corresponds to the region outside the circle with a radius of $r_s = 0.7\pi$. The filter is designed with $M = 1$, $N = 5$, $K = 4$, and $\delta = 0.07$. The magnitude response of the resulting filter is shown in Fig. 2. The designed filter is stable. The maximum pole magnitude is 0.695.

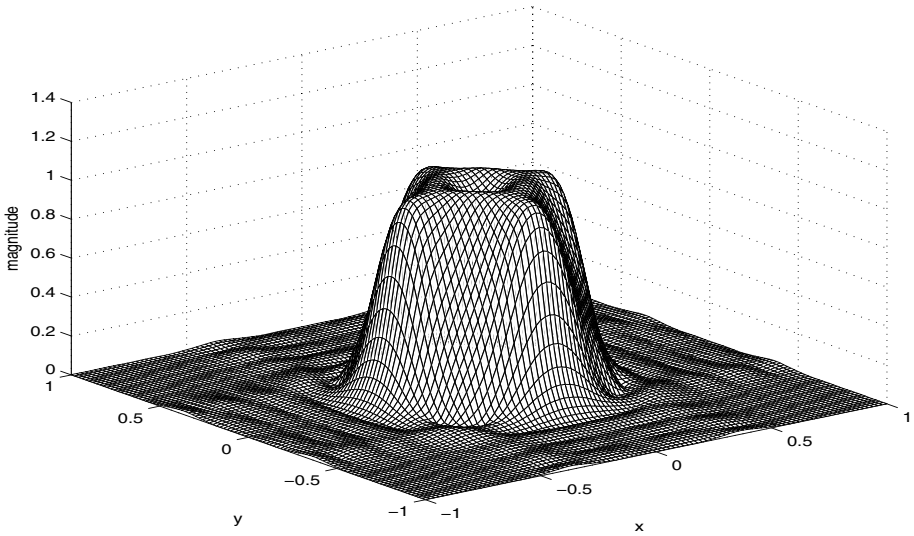


Fig. 1. Magnitude response of the filter designed in the first example ($x = \omega_1/\pi$, $x = \omega_2/\pi$)

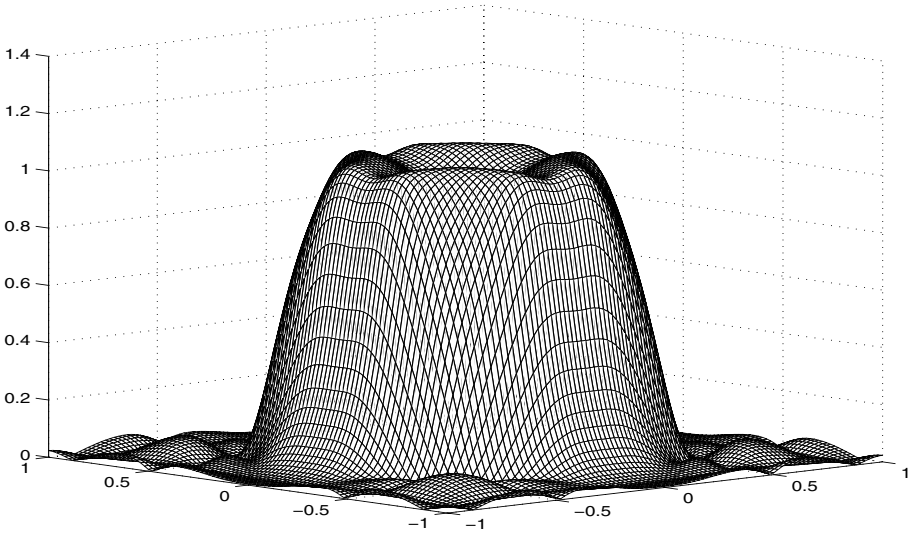


Fig. 2. Magnitude response of the filter designed in the second example ($x = \omega_1/\pi$, $x = \omega_2/\pi$)

As in case of the first example, the resulting filter can be compared with the filter obtained using the LS approach. For the LS filter, the maximum approximation error in the passband is $\delta = 0.112$. In case of the proposed approach - $\delta = 0.07$. Note that as in the previous example, in case of the proposed approach, δ is considerably smaller than in case of the LS approach.

To check usefulness of the proposed two-step solution procedure, we compare the results obtained using the proposed approach with the results obtained using a local optimization method instead of the GA in the two-step solution procedure. We design 2-D IIR filters for the same specifications as in the considered examples, but using the DFP method for solving the least square approximation problem (17). In case of the first example, the results were approximately the same as using the GA. In case of the second example, the solution of the problem (17) has been trapped at a local minimum. This local minimum is quite far from the global minimum obtained using the GA and the magnitude response of the resulting filter is not acceptable.

6 Conclusions

We have attempted to show that the GA can be used as a tool in the design of 2-D IIR filters according to the equiripple error criterion in the passband and least-squared error criterion in the stopband. A technique for the design of 2-D IIR filters with quadrantally symmetric magnitude response has been proposed. The stability of the filter is ensured by explicitly including stability constraints in the considered optimization problem. The technique is simple to implement because standard GA and local optimization procedures can be used to solve the considered minimization problem. It is also flexible as additional linear and/or nonlinear constraints can be incorporated into the optimization problem. The application of the GA ensures that the obtained solutions are not trapped at local minima. The proposed approach, in which the filter design problem is transformed into a bicriterion optimization problem, can also be applied to solving 2-D FIR filter design problems in which a compromise between the equiripple and LS errors is required [15].

References

1. Adams J. W., Sullivan J. L.: Peak-Constrained Least-Squares Optimization, *IEEE Trans. Signal Processing* **146** (1998)306–320.
2. Dumitrescu B.: Optimization of Two-Dimensional IIR Filters with Nonseparable and Separable Denominator, *IEEE Trans. on Signal Processing*, vol. 53, 2005, pp. **53** (2005) 1768–1777.
3. Goldberg D. E.: *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley, New York (1989).
4. Gorinevsky D., Boyd S., Optimization-Based Design and Implementation of Multi-Dimensional Zero-Phase IIR filters, *IEEE Trans. on Circuits and Syst.* **152** (2005) 1–12.
5. Gu Q., Swamy M. N. S.: On the Design of a Board Class of 2-D Recursive Digital Filters with Fan, Diamond and Elliptically Symmetric Responses, *IEEE Trans. Circuits and Syst.- II* **41** (1994) 603–614.
6. Karivaratharajan P., Swamy M. N. S.: Quadrantal Symmetry Associated with Two-Dimensional Transfer Functions, *IEEE Trans. on Circuits and Syst.* **25** (1978) 340–343.

7. Lim, J. S.: Two-Dimensional Signal and Image Processing. Englewood Cliffs, Prentice Hall (1990)
8. Lu W.-S., Hinamoto T.: Optimal Design of IIR Digital Filters with Robust Stability Using Conic-Quadratic-Programming Updates, *IEEE Trans. on Signal Processing* **51** (2003) 1581–1592.
9. Lu W.-S., Pei S.-C., Tseng C.-C.: A Weighted Least-Squares Method for the Design of Stable 1-D and 2-D IIR Digital Filters, *IEEE Trans. on Signal Processing* **46** (1998) 1–10.
10. Lu W.-S., Antoniou A.: Minimax Design of 2-D IIR Digital Filters Using Sequential Semidefinite Programming, *Proc. Int. Symp. Circuits Syst.* **III** (2002) 353–356.
11. Lu W.-S.: A Unified Approach for the Design of 2-D Digital Filters via Semidefinite Programming, *IEEE Trans. on Circuits and Syst. I* **45** (2002) 814–826.
12. Nocedal, J., Wright S. J.: Numerical Optimization. Springer-Verlag, Berlin Heidelberg New York (1999).
13. Reddy H. C., Khoo I-H., Rajan P. K., 2-D Symmetry: Theory and Filter Design Applications, *IEEE Circuits and Syst. Mag.* **3** (2003) 4–32.
14. Shenoj B.A. , P. Misra: Design of Two-Dimensional IIR Digital Filters with Linear Phase, *IEEE Trans. Circuits and Syst.- II* **42** (1995) 124–129.
15. Wysocka-Schillak, F.: Design of 2-D FIR Centro-Symmetric Filters With Equiripple Passband and Least-Squares Stopband. *Proc. of the Europ. Conf. on Circuit Theory and Design, Cracow, Poland* (2003) III-113–116.

A Hybrid Genetic Algorithm with Simulated Annealing for Nonlinear Blind Equalization Using RBF Networks

Soowhan Han¹, Imgeun Lee¹, and Changwook Han²

¹ Department of Multimedia Engineering, Dongeui University, Busan, Korea 614-714
{swhan, iglee}@deu.ac.kr

² School of Electrical Eng. and Computer Science, Yeungnam University, Korea 712-749
cwhan@yumail.ac.kr

Abstract. In this study, a hybrid genetic algorithm, which merges a genetic algorithm with simulated annealing, is derived for nonlinear channel blind equalization using RBF networks. The proposed hybrid genetic algorithm is used to estimate the output states of a nonlinear channel, based on the Bayesian likelihood fitness function, instead of the channel parameters. From these estimated output states, the desired channel states of the nonlinear channel are derived and placed at the center of a RBF equalizer to reconstruct transmitted symbols. In the simulations, binary signals are generated at random with Gaussian noise. The performance of the proposed method is compared with those of a conventional genetic algorithm (GA) and a simplex GA. It is shown that the relatively high accuracy and fast convergence speed have been achieved.

1 Introduction

The nonlinear inter-symbol interference (ISI) that often arises in high speed communication channels degrades the performance of the overall communication system [1]. To overcome the effects of nonlinear ISI and to achieve high-speed reliable communication, nonlinear channel equalization is necessary.

However, only a few papers have dealt with nonlinear channel models because of their complexity. The blind estimation of Volterra kernels, which characterize nonlinear channels, was derived in [2], and a maximum likelihood (ML) method implemented via expectation-maximization (EM) was introduced in [3]. The Volterra approach suffers from its enormous complexity. Furthermore the ML approach requires some prior knowledge of the nonlinear channel structure to estimate the channel parameters. Major progress in nonlinear channel blind equalization was made by Lin et al. [4], in which they used the simplex GA method to estimate the optimal channel output states instead of estimating the channel parameters directly. The desired channel states were constructed from these estimated channel output states, and placed at the center of their RBF equalizer. With this method, the complex modeling of the nonlinear channel can be avoided, and it has turned out that the nonlinear channel blind equalization problem can be transformed to the problem of determining the optimal channel output states.

In this paper, a new hybrid genetic algorithm (GA merged with simulated annealing (SA): GASA) to find optimal output states of a nonlinear channel is investigated.

GA [5] and SA [6], each of which represents a powerful optimization method, have complementary strengths and weaknesses. To get the synergy effect between GA and SA, many researchers have considered the combination of these two, and those algorithms have been successively used for the optimization problems [7][8]. Thus a hybrid architecture with GA and SA can be a possible solution to find the optimal channel output states for nonlinear channel blind equalization. For our particular application, the proposed GASA has the Bayesian fitness function in the searching procedure, and it can reach the optimal global solution with a relatively high speed even when it is trapped in a local solution. More details for its searching algorithm are shown in section 4. The performance of the proposed GASA is compared with those of a conventional GA and a simplex GA.

2 Nonlinear Channel Equalization Using RBF Networks

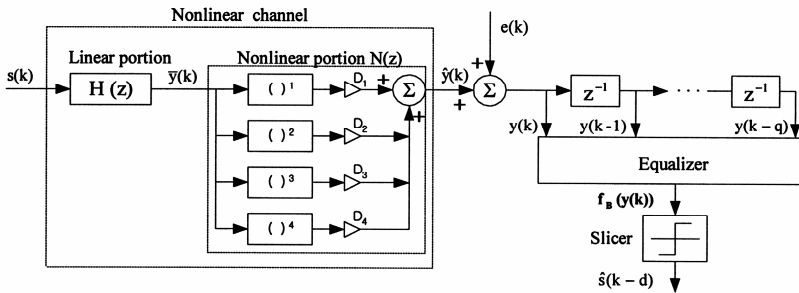


Fig. 1. The structure of a nonlinear channel equalization system

A nonlinear channel equalization system is shown in Fig. 1. A digital sequence $s(k)$ is transmitted through the nonlinear channel, which is composed of a linear portion $H(z)$ and a nonlinear portion $N(z)$, governed by the following expressions,

$$\bar{y}(k) = \sum_{i=0}^p h(i)s(k-i) \tag{1}$$

$$\hat{y}(k) = D_1\bar{y}(k) + D_2\bar{y}(k)^2 + D_3\bar{y}(k)^3 + D_4\bar{y}(k)^4 \tag{2}$$

where p is the channel order and D_i is the coefficient of the i^{th} nonlinear term. The transmitted symbol sequence $s(k)$ is assumed to be an equiprobable and independent binary sequence taking values from $\{\pm 1\}$, and the channel output is corrupted by an additive white Gaussian noise $e(k)$. Thus the channel observation $y(k)$ can be written as

$$y(k) = \hat{y}(k) + e(k) \tag{3}$$

If q denotes the equalizer order (number of tap delay elements in the equalizer), then there exist $M = 2^{p+q+1}$ different input sequences such as

$$\mathbf{s}(\mathbf{k}) = [s(k), s(k-1), \dots, s(k-p-q)] \tag{4}$$

For a specific channel order and equalizer order, the number of input patterns that influence the equalizer is M , and the input vector of equalizer without noise is

$$\hat{\mathbf{y}}(\mathbf{k}) = [\hat{y}(k), \hat{y}(k-1), \dots, \hat{y}(k-q)] \tag{5}$$

The noise-free observation vector $\hat{\mathbf{y}}(\mathbf{k})$ is referred to as the desired channel states, and can be partitioned into two sets, $\mathbf{Y}_{q,d}^{+1}$ and $\mathbf{Y}_{q,d}^{-1}$, as shown in equations (6) and (7), depending on the value of $s(k-d)$, where d is the desired time delay.

$$\mathbf{Y}_{q,d}^{+1} = \{\hat{\mathbf{y}}(\mathbf{k}) | s(k-d) = +1\} \tag{6}$$

$$\mathbf{Y}_{q,d}^{-1} = \{\hat{\mathbf{y}}(\mathbf{k}) | s(k-d) = -1\} \tag{7}$$

The task of the equalizer is to recover the transmitted symbols $s(k-d)$ based on the observation vector $\mathbf{y}(\mathbf{k})$. Because of the additive white Gaussian noise, the observation vector $\mathbf{y}(\mathbf{k})$ is a random process having conditional Gaussian density functions centered at each of the desired channel states, and determining the value of $s(k-d)$ becomes a decision problem. Therefore, Bayes decision theory [9] can be applied to derive the optimal solution for the equalizer, and this optimal Bayesian equalizer solution is given by equations (8) and (9).

$$f_B(\mathbf{y}(\mathbf{k})) = \sum_{i=1}^{n_s^{+1}} \exp\left(-\|\mathbf{y}(\mathbf{k}) - \mathbf{y}_i^{+1}\|^2 / 2\sigma_e^2\right) - \sum_{i=1}^{n_s^{-1}} \exp\left(-\|\mathbf{y}(\mathbf{k}) - \mathbf{y}_i^{-1}\|^2 / 2\sigma_e^2\right) \tag{8}$$

$$\hat{s}(k-d) = \text{sgn}(f_B(\mathbf{y}(\mathbf{k}))) = \begin{cases} +1, & f_B(\mathbf{y}(\mathbf{k})) \geq 0 \\ -1, & f_B(\mathbf{y}(\mathbf{k})) < 0 \end{cases} \tag{9}$$

where \mathbf{y}_i^{+1} and \mathbf{y}_i^{-1} are the desired channel states belonging to $\mathbf{Y}_{q,d}^{+1}$ and $\mathbf{Y}_{q,d}^{-1}$, respectively, and their numbers are denoted as n_s^{+1} and n_s^{-1} , and σ_e^2 is the noise variance. In our study, the optimal Bayesian decision probability shown in equation (8) is implemented by using a RBF network. In general, the output of a RBF network for input vector \mathbf{x} , is given by equation (10) [10].

$$f(x) = \sum_{i=1}^n \omega_i \phi\left(\frac{\|\mathbf{x} - \mathbf{c}_i\|^2}{\rho_i}\right) \tag{10}$$

where n is the number of hidden units, \mathbf{c}_i are the RBF centers, ρ_i is the width of the i^{th} units and ω_i is its weight. The RBF network is an ideal processing means to implement the optimal Bayesian equalizer when the nonlinear function ϕ is chosen as the exponential function $\phi(x) = e^{-x}$ and all of the widths have a same value ρ , which is twice as large as the noise variance σ_e^2 . For the case of equiprobable symbols, the RBF network can be simplified by setting half of the weights to 1 and the other half to -1. Thus the output of this RBF equalizer in equation (10) is same as the optimal Bayesian decision probability shown in equation (8).

3 Desired Channel States and Channel Output States

The desired channel states, \mathbf{y}_i^{+1} and \mathbf{y}_i^{-1} , are used as the centers of the hidden units in the RBF equalizer to reconstruct the transmitted symbols. If the channel order $p = 1$ with $H(z) = 0.5 + 1.0z^{-1}$, the equalizer order $q = 1$, the time delay $d = 1$, and the nonlinear portion $D_1 = 1, D_2 = 0.1, D_3 = 0.05, D_4 = 0.0$ in Fig. 1, then the eight different channel states ($2^{p+q+1} = 8$) may be observed at the receiver in the noise-free case, and the output of the equalizer should be $\hat{s}(k-1)$, as shown in Table 1. From Table 1, it can be seen that the desired channel states $[\hat{y}(k), \hat{y}(k-1)]$ can be constructed from the elements of the dataset, called ‘‘channel output states’’, $\{a_1, a_2, a_3, a_4\}$, where $a_1 = 1.89375, a_2 = -0.48125, a_3 = 0.53125, a_4 = -1.44375$. The length of dataset, \tilde{n} , is determined by the channel order, p , such as $2^{p+1} = 4$. This relation is always valid for the channel that has a one-to-one mapping between the channel inputs and outputs [4]. Thus the desired channel states can be derived from the channel output states if we assume p is known, and the main problem of blind equalization can be changed to focus on finding the optimal channel output states from the received patterns.

Table 1. The relation between desired channel states and channel output states

| Nonlinear channel with $H(z) = 0.5 + 1.0z^{-1}, D_1 = 1, D_2 = 0.1, D_3 = 0.05, D_4 = 0.0, d=1$ | | | | | | |
|---|----------|----------|------------------------|----------------|---|----------------|
| Transmitted symbols | | | Desired channel states | | Output of equalizer | |
| $s(k)$ | $s(k-1)$ | $s(k-2)$ | $\hat{y}(k)$ | $\hat{y}(k-1)$ | By channel output states, $\{a_1, a_2, a_3, a_4\}$ | $\hat{s}(k-1)$ |
| 1 | 1 | 1 | 1.89375 | 1.89375 | (a_1, a_1) | 1 |
| 1 | 1 | -1 | 1.89375 | -0.48125 | (a_1, a_2) | 1 |
| -1 | 1 | 1 | 0.53125 | 1.89375 | (a_3, a_1) | 1 |
| -1 | 1 | -1 | 0.53125 | -0.48125 | (a_3, a_2) | 1 |
| 1 | -1 | 1 | -0.48125 | 0.53125 | (a_2, a_3) | -1 |
| 1 | -1 | -1 | -0.48125 | -1.44375 | (a_2, a_4) | -1 |
| -1 | -1 | 1 | -1.44375 | 0.53125 | (a_4, a_3) | -1 |
| -1 | -1 | -1 | -1.44375 | -1.44375 | (a_4, a_4) | -1 |

It is known that the Bayesian likelihood (BL), defined in equation (11), is maximized with the channel states derived from the optimal channel output states [11].

$$BL = \prod_{k=0}^{L-1} \max(f_B^{+1}(k), f_B^{-1}(k)) \tag{11}$$

where

$$f_B^{+1}(k) = \sum_{i=1}^{n_+^{+1}} \exp\left(-\|\mathbf{y}(k) - \mathbf{y}_i^{+1}\|^2 / 2\sigma_c^2\right), \quad f_B^{-1}(k) = \sum_{i=1}^{n_+^{-1}} \exp\left(-\|\mathbf{y}(k) - \mathbf{y}_i^{-1}\|^2 / 2\sigma_c^2\right)$$

and L is the length of received sequences. Therefore, the BL is utilized as the fitness function (FF) of the proposed algorithm to find the optimal channel output states after taking the logarithm. However, the optimal channel output states, which maximize FF , cannot be obtained with the conventional gradient methods, because the mathematical formulation between the channel output states and FF cannot be accomplished without knowing the channel structure. These are shown in [4]. Thus a new hybrid genetic algorithm, called GASA, is applied to search for the optimal output states, and is compared with a conventional GA and the simplex GA introduced in [4].

4 GA Merged with SA (GASA)

GA and SA have complementary strengths and weaknesses. While GA explores the search space by means of the population of search points, it suffers from poor convergence properties. SA, by contrast, has good convergence properties, but it cannot explore the search space by means of population. To get the synergy effect between GA and SA, a new hybrid genetic algorithm, which combines GA with SA, is investigated and applied to find the optimal channel output states for nonlinear channel blind equalization. The proposed GASA algorithm has the following pseudo-code. The Bayesian likelihood in equation (11) is utilized as the fitness function, and thus GASA searches the channel output states which maximize the Bayesian likelihood.

```

begin
    initialize population at random
    randomly generate the initial temperature T[i] in
    a specified region
    calculate the fitness function for the initial
    population
    for k=1 to stopping criterion
        begin
            save the current population as parents
            crossover
            mutation
            find the best-fit individual among the
            parents and offsprings and then update the
            best solution
            for i=1 to population size
                begin
                    ith-individual=SA-selection(SA-selection(offspring[i],
                    parent[i],T[i]),
                best solution, T[i])
                    update the ith-individual fitness

```

```

        T[i]=T[i]×cooling rate
    end
end
end
end

```

In the pseudo-code, the selection of SA, shown in “SA-selection(SA-selection(offspring[i], parent[i], T[i]), best solution, T[i])”, has been modified to have its fitness function maximized. For example, the function “SA-selection(new, old, T)” calculates the acceptance probability “ $P = \exp(-(\text{old} - \text{new}) / T)$ ”. If “ $\text{new} > \text{old}$ ”, a “new” solution is selected, which means that an uphill move is always accepted. And also, if “ $\text{new} \leq \text{old}$ ” and “ $P > \text{random number in } [0, 1]$ ”, a “new” solution will be selected, which means that a downhill move is occasionally accepted, depending on P . An “old” solution will be selected for all other cases. This selection of SA allows a downhill move (same as an uphill move in a typical SA which minimizes the fitness function) to explore the search space at higher temperature, and to exploit the search space acceptance of the best solution’s individual at lower temperatures. Thus in this algorithm, the GA-selection is effectively replaced with an SA-selection without increasing the number of fitness evaluations per generation. This means that the population stores a diversity of annealing schedules, and the proposed GASA can reach the optimum global solution with a relatively high speed even when it is trapped in a local solution.

5 Simulation Results and Performance Assessments

The blind equalizations with GA, simplex GA, and GASA are taken into account to show the effectiveness of the proposed hybrid algorithm. Two nonlinear channels in [4] and [12] are evaluated in the simulations. Channel 1 is shown in Table 1, and the other is as follows.

Channel 2:

$$H(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2},$$

$$D_1 = 1, D_2 = 0.2, D_3 = 0.0, D_4 = 0.0 \text{ and } d = 1$$

In channel 2, the channel order p , the equalizer order q , and the time delay d are 2, 1, 1, respectively. Thus the output of the equalizer should be $\hat{s}(k-1)$, and the sixteen desired channel states ($2^{p+q+1} = 16$) composed of the eight channel output states ($2^{p+1} = 8, a_1, a_2, a_3, \dots, a_8$) may be observed at the receiver in the noise-free case. The desired channel states, $(a_1, a_1), (a_1, a_2), (a_2, a_3), (a_2, a_4), (a_5, a_1), (a_5, a_2), (a_6, a_3), (a_6, a_4)$, belong to $Y_{1,1}^{+1}$, and $(a_3, a_5), (a_3, a_6), (a_4, a_7), (a_4, a_8), (a_7, a_5), (a_7, a_6), (a_8, a_7), (a_8, a_8)$ belong to $Y_{1,1}^{-1}$, where $a_1, a_2, a_3, \dots, a_8$ are 2.0578, 1.0219, -0.1679, -0.7189, 1.0219, 0.1801, -0.7189 and -1.0758, respectively.

In the experiments, 10 independent simulations for each of two channels with five different noise levels (SNR=5,10,15,20 and 25db) are performed with 1000 randomly generated transmitted symbols, and the results are averaged. The three algorithms,

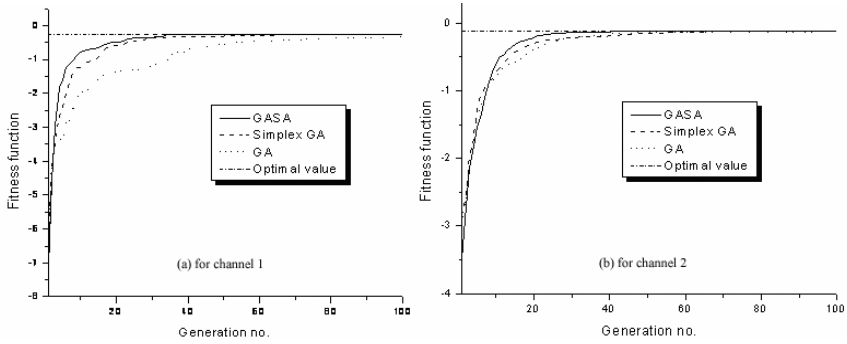


Fig. 2. Averaged fitness functions in successive 100 generations: (a) channel 1, (b) channel 2

GA, simplex GA and proposed GASA, have been implemented in a batch way in order to obtain an accurate comparison among them, and the same quantities of population size, crossover rate, and mutation rate are used. The averaged fitness functions in successive generations with 25db are shown in Fig. 2 for each of the two channels. It is observed that the proposed GASA converges with the highest speed because of its diversity of annealing schedules as mentioned in the previous section.

We also measure the normalized root mean squared errors (NRMSE) for the estimation of channel output states, defined by equation (12), and they are shown in Fig. 3. GASA presents the lowest NRMSE over all of the SNR ranges. A sample of 1000 received symbols under 5db SNR for channel 2 and their desired channel states constructed from the estimated channel output states by GASA is shown in Fig. 4.

$$NRMSE = \frac{1}{\|a\|} \sqrt{\frac{1}{m} \sum_{i=1}^m \|a - \hat{a}_i\|^2} \tag{12}$$

where a is the dataset of optimal channel output states, \hat{a}_i is the dataset of estimated channel output states, and m is the number of simulations performed ($m = 10$). Finally, the bit error rates (BER) are checked and summarized in Table 2. It is shown that the BER, with the estimated channel output states by GASA, is almost same as the one with the optimal output states for both of channel 1 and 2.

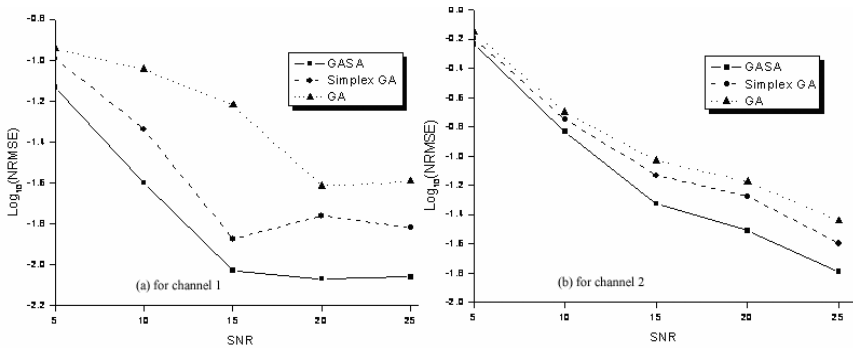


Fig. 3. NRMSE: (a) channel 1, (b) channel 2

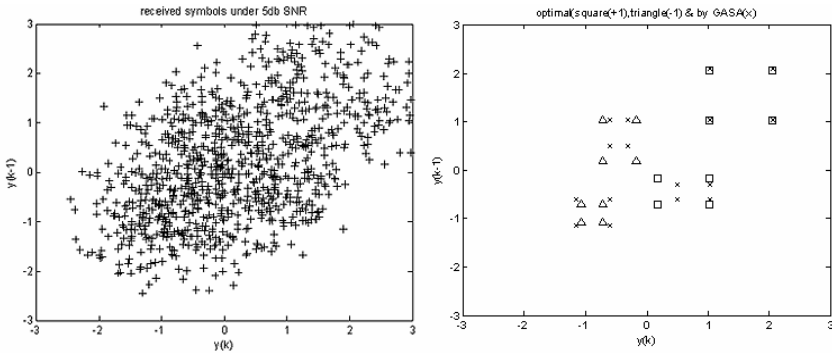


Fig. 4. A sample of received symbols for channel 2 and their desired channel states by GASA

Table 2. Averaged BER(no. of errors/no. of transmitted symbols) for channel 1 and 2

| Channel | SNR | with optimal states | GASA | Simplex GA | GA |
|-----------|------|---------------------|--------|------------|--------|
| Channel 1 | 5db | 0.0797 | 0.0815 | 0.0824 | 0.0816 |
| | 10db | 0.0121 | 0.0120 | 0.0128 | 0.0136 |
| | 15db | 0 | 0 | 0 | 0.0003 |
| | 20db | 0 | 0 | 0 | 0 |
| | 25db | 0 | 0 | 0 | 0 |
| Channel 2 | 5db | 0.0970 | 0.1070 | 0.1162 | 0.1210 |
| | 10db | 0.0420 | 0.0460 | 0.0492 | 0.0502 |
| | 15db | 0.0100 | 0.0112 | 0.0114 | 0.0112 |
| | 20db | 0.0008 | 0.0008 | 0.0008 | 0.0008 |
| | 25db | 0 | 0 | 0 | 0 |

6 Conclusions

A new genetic algorithm merged with SA (GASA) is presented for nonlinear channel blind equalization. In this study, the complex modeling of an unknown nonlinear channel becomes unnecessary by constructing the desired channel states directly from the estimated channel output states. It has been shown that the proposed GASA with the Bayesian likelihood as the fitness function offers better performance than conventional GA and simplex GA. It successively estimates the channel output states with relatively high speed and accuracy. Thus a RBF equalizer, based on GASA, can be a possible solution for nonlinear blind channel equalization problems.

References

1. Biglieri, E., Gersho, A., Gitlin, R. D., Lim, T. L.: Adaptive cancellation of nonlinear inter-symbol interference for voiceband data transmission. *IEEE J. Selected Areas Commun.* SAC-2(5) (1984) 765-777

2. Stathaki, T., Scohyers, A.: A constrained optimization approach to the blind estimation of Volterra kernels. *Proc. IEEE Int. Conf. on ASSP 3* (1997) 2373-2376
3. Kaleh, G. K., Vallet, R.: Joint parameter estimation and symbol detection for linear or nonlinear unknown channels. *IEEE Trans. Commun.* 42 (1994) 2406-2413
4. Lin, H., Yamashita, K.: Hybrid simplex genetic algorithm for blind equalization using RBF networks. *Mathematics and Computers in Simulation* 59 (2002) 293-304
5. Goldberg, D. E.: *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, Reading, MA (1989)
6. Romeo, F., Sangiovanni-Vincentelli, A.: A theoretical framework for simulated annealing. *Algorithmica* 6 (1991) 302-345
7. Li, B., Jiang, W.: A novel stochastic optimization algorithm. *IEEE Trans. SMC-B*(30) (2000) 193-198
8. Savchenko, V., Schmitt, L.: Reconstructing occlusal surfaces of teeth using a genetic algorithm with simulated annealing type selection. *Proc. of the sixth ACM symposium on Solid modeling and applications, Michigan, U.S.A.* (2001) 39-46
9. Duda, R. O., Hart, P. E.: *Pattern Classification and Scene Analysis*. New York, Wiley(1973)
10. Lee, J., Beach, C., Tepedelenlioglu, N.: Channel Equalization using Radial Basis Function Network. *ICASSP, Atlanta, Georgia, U.S.A.* 3 (1996) 1719-1722
11. Lin, H., Yamashita, K.: Blind equalization using parallel Bayesian decision feedback equalizer. *Mathematics and Computers in Simulation* 56 (2001) 247-257
12. Patra, S. K., Mulgrew, B.: Fuzzy techniques for adaptive nonlinear equalization. *Signal Process* 80 (2000) 985-1000

Feature Extraction of Speech Signal by Genetic Algorithms-Simulated Annealing and Comparison with Linear Predictive Coding Based Methods

Melih İnal

Kocaeli University, Technical Education Faculty, Electronics and Computer Department,
Umuttepe Campus, 41380, İzmit, Kocaeli, Turkey
minal@kou.edu.tr

Abstract. This paper presents Genetic Algorithms and Simulated Annealing (GASA) based on feature extraction of speech signal and comparison with traditional Linear Predictive Coding (LPC) methods. The performance of each method is analyzed for ten speakers with independent text speaker verification database from Center for Spoken Language Understanding (CSLU) which was developed by Oregon Graduate Institute (OGI). The GASA algorithm is also analyzed with constant population size for different generation numbers, cross-over and mutation probabilities. When compared with the Mean Squared Error (MSE) of the each speech signal for each method, all simulation results of the GASA algorithm are more effective than LPC methods.

1 Introduction

Feature extraction is the most important stage of the speech processing. Speech signals should be analyzed by effective methods. Then feature vectors obtained from these analyzed speech signals are used to form a robust speech processing system. LPC is an analytically tractable model. The methods of LPC are mathematically precise and straightforward to implement.

LPC models the signal as a linear combination of its past values. In the frequency domain, this is equivalent to modeling the signal spectrum by a pole-zero spectrum [1]. The analysis of the speech signal is concern of the “time series analysis”. Each continuous-time signal $s(t)$ is sampled to obtain a discrete-time signal $s(n)$, also known as time series. A major concern of time series analysis is the estimation of power spectra, autocorrelation and cross-correlation functions. It is clear that if one is successful in developing a parametric model for the behavior of some signal, then the model can be used for different applications, such as recognition, prediction or forecasting, control, and data compression. The following equation is one of the most powerful models [2]. In (1), a_i ($1 \leq i \leq p$; p is degree of the parameter set) are LPC coefficients, the excitation gain factor G and excitation source $u(n)$ are the parameters of the system.

The excitation function is either a quasi-periodic impulse sequence for the voiced sounds or a random noise source for unvoiced sounds with a gain factor G set to control the intensity of the excitation.

$$s(n) = \sum_{i=1}^p a_i s(n-i) + Gu(n) \tag{1}$$

The gain factor G is usually ignored to allow the parameterization to be independent of the signal intensity. Equation (1) shows that the signal $s(n)$ is predictable from linear combinations of past outputs and inputs. Hence, the name of the algorithm is linear predictive coding. The LPC coefficients a_i can be derived from Autocorrelation and Covariance based methods [3]. Moreover, Cepstral Coefficients (c_i) can be derived from LPC coefficients a_i as follows [4]:

$$c_1 = a_1$$

$$c_n = \sum_{k=1}^{n-1} (1 - k/n) a_k c_{n-k} + a_n \tag{2}$$

Also optimum prediction of the signal $s(n)$ can be done by genetic algorithms (GA). Genetic algorithms are global search techniques for optimization [5], [6]. However, they are poor at hill-climbing [7]. Simulated annealing (SA) has the ability of probabilistic hill-climbing. Therefore, hybridization of two techniques which is called GASA supplements each other by their special abilities.

Recently published numerous studies apply genetic and simulated annealing algorithms for different application areas such as photovoltaic power systems, digital filters, unit commitment, power system operation, load forecasting and manufacturing [8]-[13].

2 Proposed Algorithm

The flowchart of the proposed GASA algorithm is shown in Fig. 1. GASA parameter settings and operations are also tabulated in Table 1. $Nsame$ parameter shows how many generations the fittest chromosome is unchanged. In Table 1, each parameter except for population size (Ps) and $Nsame$ is analyzed separately keeping all others fixed. Ps and $Nsame$ parameters are chosen 50 and 1000 respectively for all the others. Therefore, proposed algorithm works like a simple GA by reducing the SA feature of GASA. Then, different values, such as 10, 100 and 500, of $Nsame$ parameters are only applied to the best result of the GASA parameter setting. Because of LPC coefficients are floating points, chromosome type is chosen real coded.

Determining initial temperature T_0 is generally difficult because it depends on the problem being solved. T_0 is defined as the fitness value of the worst chromosome in the population. In the simulation, cooling schedule for simulated annealing process is defined as follows:

$$T(k) = \frac{\log(k)}{T_0} \tag{3}$$

where $k=1, 2, \dots, N$ and N is generation number.

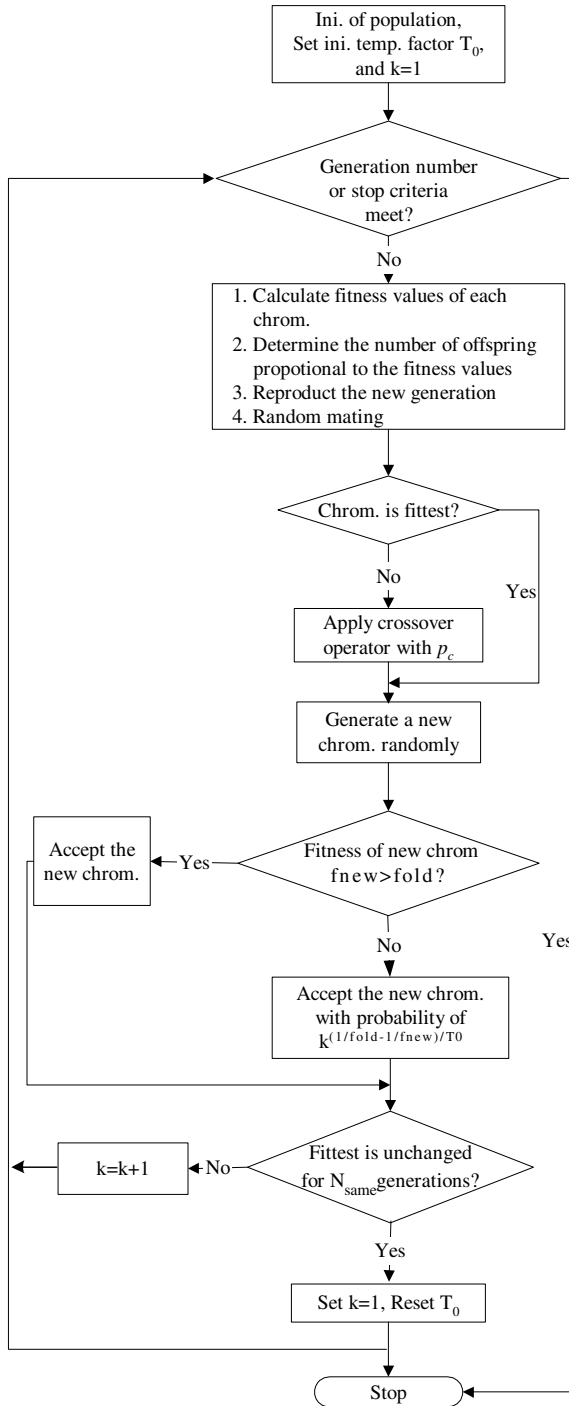


Fig. 1. The flowchart of the proposed GASA algorithm

The hybrid algorithm GASA is proposed here to improve the performance of the GA. In general, SA searches for the minimal energy state while GA searches for the chromosome with the maximum fitness value. No change is made to the crossover operator for preserving the accomplishment of GA. The proposed GASA algorithm includes the merits of SA by changing the mutation operator. The new mutation which is basically different only from changing the operator probability operates like SA as follows: if a random generated new chromosome is better than the original one, it accepts. Otherwise, the new chromosome is accepted according to the probability given as follows:

$$p(k) = \exp \left[- \left(\frac{1}{F_{new}} - \frac{1}{F_{old}} \right) * T(k) \right], \tag{4}$$

where F_{old} is the fitness value of the original chromosome and F_{new} is the fitness value of the randomly generated chromosome.

From (3) and (4) the probability can be written as follows:

$$p(k) = \exp \left[- \left(\left(\frac{1}{F_{new}} - \frac{1}{F_{old}} \right) * \log(k) \right) / T_0 \right] = k^{\left(\frac{1}{F_{old}} - \frac{1}{F_{new}} \right) / T_0}. \tag{5}$$

The simple elitism strategy, that is always transferring the best solution to the next generation without any alterations, is applied in order to get a good solution under the condition of randomized search due to the repeated cooling schedules.

Each speech frame overlapped and framed by Hamming window is processed in GASA algorithm for estimating the each sample of $s(n)$. The initial population is used here as prediction coefficients. After the total error TE of the framed speech which has length of N_i is calculated, the F fitness value of the k -th population can be calculated as follows:

$$F(k) = 1 / \sqrt{TE / N_i}. \tag{6}$$

The modified mutation operator increases hill-climbing capability of GA. From the perspective of GA, the modified mutation operator functions as the standard one does. From the perspective of SA, it searches multiple neighborhood points that are presented for the selection of the fittest value of the chromosome.

3 Experiments and Results

The speech data for 10 speakers from CSLU speaker verification database are used in the experiments. Text independent data was pronounced by 5 female and 5 male speakers. The female and male speakers are labeled as F1-F5 and M1-M5 respectively. The preprocessing of the speech data consists of several steps. The speech data, which is sampled at 8 KHz, is processed by the application of a pre-emphasis filter $H(z)=1-0.98z^{-1}$. The 40-ms Hamming window is applied to the speech every

20 ms. 12th-order LPC analysis is performed for each speech frame. First, Auto-correlation and Covariance analyses are performed for determining LPC coefficients. Then c_i are derived from a_i coefficients that are obtained from Autocorrelation analysis according to (2). All these feature vectors are applied to the framed speech signal for determining the Mean Squared Errors (MSEs). Also, the same speech frames are used to determine 12 coefficients with GASA algorithm which is performed by (1), (3), (4), (5) and (6).

GASA experiments are realized according to the initial parameters and operations which are shown in Table 1. Since the degree of the parameter set p is 12, the population of GA is initialized randomly between -1 and +1 in a size of 50x12. The same initial population is used for all simulations.

Table 1. GASA initial parameters and operations

| | |
|---------------------------|---------------------------------|
| Number of Generation | $N = 500, 1000$ |
| Population size | $Ps = 50$ |
| Chromosome length | 12 |
| Chromosome type | Real-coded |
| Mutation probability | $Pm = 0.01, 0.05, 0.1$ |
| Crossover probability | $Pc = 0.6, 0.8, 1$ |
| Selection type | Roulette well |
| Mutation type | Chromosome by chromosome |
| Crossover type | One cut point |
| Elitism | Simple |
| Nsame generations | 10, 100, 500, 1000 |
| Initial temperature T_0 | $1/\min(\text{fitness values})$ |

Coefficients of each speaker's speech frame are computed by GASA algorithm for different parameter settings. Then results of MSEs are computed according to these coefficients. Table 2 shows the results of MSEs for different parameter settings. After each line of generation number in Table 2, minimum, maximum and mean values of the MSEs are computed and minimum of them are indicated by bold characters in related column. According to these three lines, the simulation result of 1000 generation number, 0.8 crossover and 0.1 mutation probabilities with 1000 N_{same} value for 50 generation number the best convergence result is obtained. N_{same} simulations are started for examining the effect of SA according to these parameters of the best convergence of MSE results of Table 2.

The parameter N_{same} is changed for 10, 100 and 500 values to avoid repetitiveness of GA. These results are shown in Table 3. The MSE values of each speaker appear the same for the simulation result of N_{same} parameter value of 500 and value of 1000. But in some values, there is a difference between them after 7 floating points.

Table 2. The results of MSEs for different parameter settings

| | | | | | | | | | |
|-------------|------|------|-------------|------|------|-------------|------|------|------|
| Ps | 50 | | | | | | | | |
| Pc | 0.6 | | | 0.8 | | | 1 | | |
| Pm | 0.01 | 0.05 | 0.1 | 0.01 | 0.05 | 0.1 | 0.01 | 0.05 | 0.1 |
| N | 500 | | | | | | | | |
| Min. (1E-6) | 6.41 | 4.50 | 4.12 | 7.11 | 4.98 | 4.31 | 7.72 | 5.02 | 4.35 |
| Max. (1E-5) | 5.47 | 3.67 | 3.22 | 8.08 | 3.92 | 3.41 | 6.47 | 3.91 | 3.27 |
| Mean (1E-5) | 2.45 | 1.61 | 1.40 | 3.01 | 1.74 | 1.46 | 2.77 | 1.77 | 1.47 |
| N | 1000 | | | | | | | | |
| Min. (1E-6) | 3.83 | 3.94 | 3.71 | 6.14 | 4.06 | 3.70 | 6.73 | 4.25 | 3.83 |
| Max. (1E-5) | 2.92 | 3.07 | 2.92 | 6.31 | 3.17 | 2.89 | 5.35 | 3.28 | 2.92 |
| Mean (1E-5) | 1.29 | 1.34 | 1.25 | 2.42 | 1.38 | 1.23 | 2.36 | 1.47 | 1.29 |

Table 3. The effect of Nsame parameter on MSEs

| | | | | | | |
|----------------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|
| Generation Size (N) | | 1000 | | | | |
| Population Size (Ps) | | 50 | | | | |
| Crossover Probability (Pc) | | 0.8 | | | | |
| Mutation Probability (Pm) | | 0.1 | | | | |
| Nsame | | 10 | 100 | 500 | 1000 | Minimum |
| LABELLED SPEAKERS | F1 | 2,84778362E-05 | 2,87501039E-05 | 2,88906904E-05 | 2,88906904E-05 | 2,84778362E-05 |
| | F2 | 3,62995819E-06 | 3,67193943E-06 | 3,70459163E-06 | 3,70459165E-06 | 3,62995819E-06 |
| | F3 | 3,83697018E-06 | 3,83029643E-06 | 3,83417533E-06 | 3,83417533E-06 | 3,83029643E-06 |
| | F4 | 6,37711413E-06 | 6,31526953E-06 | 6,34126715E-06 | 6,34126715E-06 | 6,31526953E-06 |
| | F5 | 4,17349989E-06 | 4,14707818E-06 | 4,14257726E-06 | 4,14257739E-06 | 4,14257726E-06 |
| | M1 | 2,44103531E-05 | 2,42073795E-05 | 2,40697917E-05 | 2,40697917E-05 | 2,40697917E-05 |
| M2 | 1,63620051E-05 | 1,63924172E-05 | 1,64936895E-05 | 1,64936897E-05 | 1,63620051E-05 | |
| M3 | 1,68991610E-05 | 1,68412918E-05 | 1,68554671E-05 | 1,68554671E-05 | 1,68412918E-05 | |
| M4 | 8,65258770E-06 | 8,44108915E-06 | 8,46801982E-06 | 8,46801645E-06 | 8,44108915E-06 | |
| M5 | 1,09287557E-05 | 1,06404238E-05 | 1,06330561E-05 | 1,06330561E-05 | 1,06330561E-05 | |
| Mean | | 1,23748241E-05 | 1,23237289E-05 | 1,23433326E-05 | 1,23433323E-05 | 1,23237289E-05 |

When examining the MSEs result of *Nsame* parameter such as 500 and 1000, it is clear that even 500 value of *Nsame* parameter is quite high for optimal SA procedure. Table 3 shows that the *Nsame* parameter with value 100 gives the best result for SA procedure. Thereby, SA procedure with proper *Nsame* value such as 100 assists the GA for hill climbing.

The best result of GASA is chosen from Table 2 and Table 3 to compare with traditional LPC methods of Autocorrelation, Covariance and Cepstral Coefficients. Comparison results of these methods are given in Table 4. It is proven that Covariance method, again, gives better results than Autocorrelation method. Also the

Cepstral Coefficients method which is the dominant method in speech processing outperforms the LPC based methods as expected. But prediction performance of GASA algorithm is more robust than the others as is seen from last row of Table 4. The average value of MSE result of the GASA algorithm is almost $2.9E-4$ point less than the other methods.

Table 4. Comparison of LPC based methods and GASA Algorithm

| Speakers | Autocorrelation | Covariance | Cepstral Coefficients | GASA |
|-------------|-----------------|------------|-----------------------|------------|
| F1 | 6.3937E-04 | 6.3867E-04 | 6.4643E-04 | 2.8750E-05 |
| F2 | 1.1652E-04 | 1.1649E-04 | 1.0352E-04 | 3.6719E-06 |
| F3 | 4.9340E-05 | 4.9249E-05 | 4.0595E-05 | 3.8303E-06 |
| F4 | 1.9082E-04 | 1.9038E-04 | 2.4626E-04 | 6.3153E-06 |
| F5 | 7.5049E-05 | 7.4929E-05 | 5.2227E-05 | 4.1471E-06 |
| M1 | 5.8750E-04 | 5.8755E-04 | 6.3220E-04 | 2.4207E-05 |
| M2 | 2.4346E-04 | 2.4316E-04 | 1.5136E-04 | 1.6392E-05 |
| M3 | 2.8805E-04 | 2.8780E-04 | 2.6460E-04 | 1.6841E-05 |
| M4 | 4.8009E-04 | 4.7965E-04 | 5.7291E-04 | 8.4411E-06 |
| M5 | 4.2526E-04 | 4.2499E-04 | 2.2277E-04 | 1.0640E-05 |
| Mean | 3.0955E-04 | 3.0929E-04 | 2.9329E-04 | 1.2324E-05 |

Since the effect of *Nsame* parameter is indicated in Table 3, the MSE results vs. each speaker and mean values are given in Fig. 2 as subplots for the best GA's parameters such as generation number, crossover and mutation probabilities. The general opinion can be repeated except for the first female speech data: "Speech data of females are more distinguishable than males' speech data". As it is seen from Fig.2 (a), (b) and (c); 1000 generation number, 0.8 crossover and 0.1 mutation probabilities give the optimum values of GASA parameters for the average values of MSEs. The results of speaker F2 are the best MSE values when MSE results of the labeled speakers are compared as shown in Fig. 2. According to the best results of the MSE of speaker F2's speech signal, using the 43rd middle frame's max, min and mean fitness values and the coefficients of LPC computed with GASA algorithm are plotted as shown in figure 3. In Fig. 3 (a), the coefficients have small changes after 600 generations, and become stable about 900th generation. It can be seen in Fig. 3 (b), fitness values increase for each generation. That means, while the total prediction error of each speech frame decreases, the fitness value increases for each generation. So, the proposed GASA algorithm properly converges according to object function which is determined by means of (6).

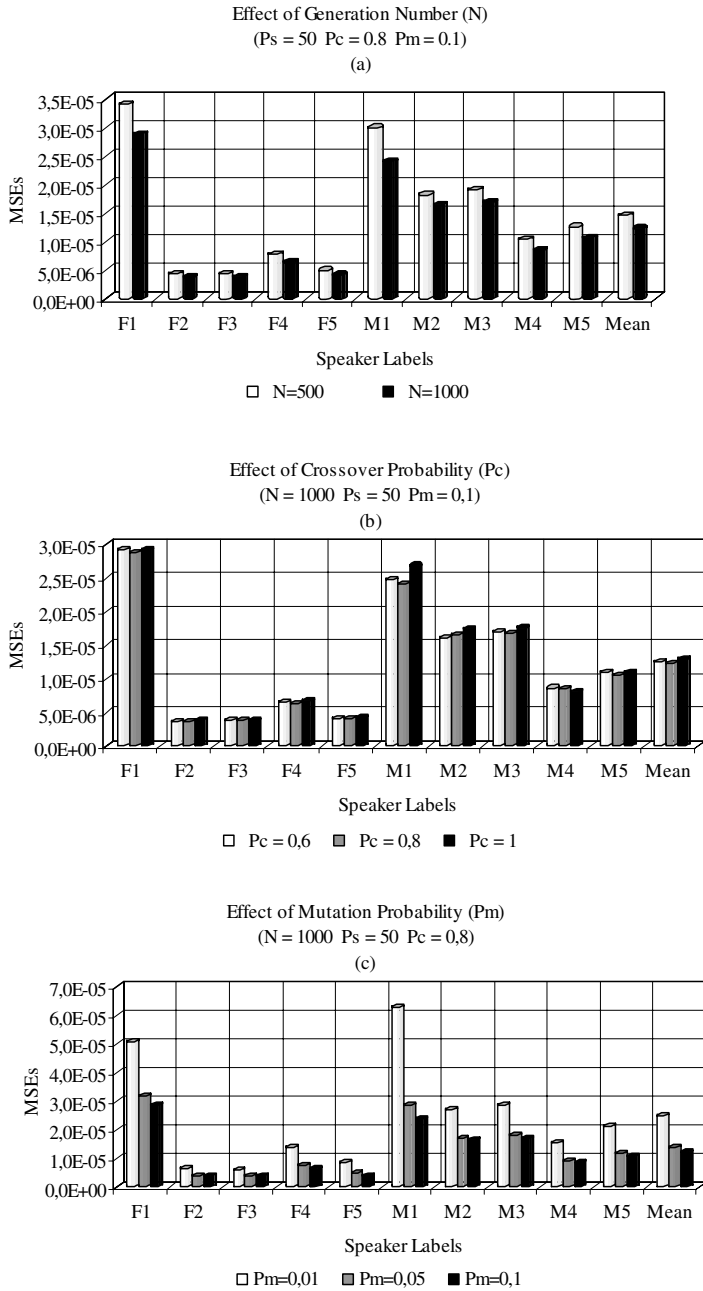


Fig. 2. The MSE results vs. each speaker and mean values: (a) Generation number, (b) Cross-over probability, (c) Mutation probability

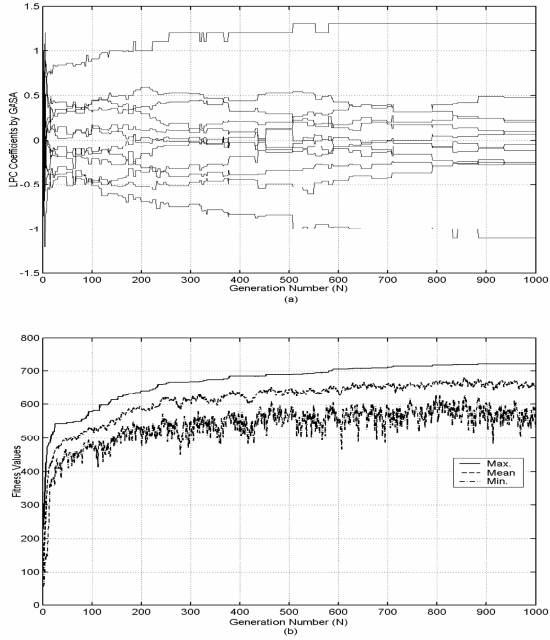


Fig. 3. The 43rd frame of Speaker F2: (a) Generation Number (N) vs. LPC coefficients computed by GASA, (b) Generation Number (N) vs. max., mean and min. fitness values

4 Conclusion

Comparison of LPC based methods and GASA algorithm is done for prediction of discrete signal $s(n)$ from its past values. The simulation result of 1000 generation number, 0.8 crossover and 0.1 mutation probabilities with 1000 N_{same} value for 50 generation number gives the best result of GASA algorithm. Since generation number and N_{same} parameters are 1000, the simulation result shows the effect of GA. Then N_{same} parameter is changed to 10, 100 and 500 values in different simulations to determine the effect of SA according to the mentioned parameters of the best MSE results. When N_{same} parameter is chosen 100, SA assists the GA for hill climbing. Therefore GA does not suffer from repetitiveness. While the fitness values increase total error decreases for each generation. This situation shows that proposed GASA algorithm properly converges. The best result of GASA with 100 value of N_{same} parameter is compared with traditional LPC methods of Autocorrelation, Covariance and Cepstral Coefficients. Comparison results of MSEs of these methods show that GASA algorithm is almost 10^{-4} point less than the other methods. When compare the MSEs of the each speech signal for each method, all simulation results of the GASA algorithm are more effective than LPC methods. GASA algorithm provides 0.1 more accuracy than the others. That means discrete signal $s(n)$ is estimated 10% more precisely with the GASA algorithm. Therefore, this improvement affects positively the

recognition tasks. Instead of simple elitism, different elitism and evolutionary optimization tasks such as Particle Swarm Optimization (PSO) [14] are proposed for future studies.

Acknowledgments. The author would like to thank the Center for Spoken Language Understanding, Oregon Graduate Institute, for supplying the Speaker Recognition V1.1 speech database.

References

1. Haykin. S.: Adaptive Filter Theory. Prentice-Hall Int. Editions. (1991)
2. Makhoul. J.: Linear Prediction: A Tutorial Review. Proc. IEEE. Vol. 63. (1975) 561-580
3. Deller. J. R., Proakis. J.G., Hansen. J. H. L.: Discrete-Time Processing of Speech Signal. Macmillan Pub. Co. (1993)
4. Furui. S.: Cepstral Analysis Technique for Automatic Speaker Verification. IEEE Trans. On Acoustics Speech and Signal Processing. Vol. ASSP-29. (1981) 254-272
5. Man. K.F., Tang. K. S. and Kwong. S.: Genetic Algorithms: Concepts and Applications. IEEE Trans. Industrial Electronics. Vol. 43. (1996) 519-533
6. Vasconcelos. J. A., Ramirez. J. A., Takahashi. R. H. C. and Saldanha. R. R.: Improvements in Genetic Algorithms. IEEE Trans. Magnetics. Vol. 37. (2001) 3414-3417
7. Prügel-Bennett. A.: When a Genetic Algorithm Outperforms Hill-Climbing. Theoretical Computer Science. Vol. 320. (2004) 135-153
8. Chen. Y.-M., Lee. C.-H., Wu. H.-C.: Calculation of the Optimum Installation Angle for Fixed Solar-Cell Panels Based on the Genetic Algorithm and the Simulated-Annealing Method. IEEE Trans. On Energy Conversion. Vol. 20. (2005) 467-473
9. Haseyama. M. and Matsuura. D.: A Filter Coefficients Quantization Method With Genetic Algorithms. Including Simulated Annealing. IEEE Signal Processing Letters. Vol. 13. (2006) 189-192
10. Sailesh Babu. G.S., Bhagwan Das. D. and Patvardhan. C.: A Hybrid Stochastic Search Approach for Unit Commitment with Hard Reserve Constraints. IEEE Power India Conf., (2006) 355-362
11. Liao. G.-C.: Application of an immune algorithm to the short-term unit commitment problem in power system operation. IEE Proc.-Gener. Transm. Dist. Vol. 153. (2006) 309-320
12. Liao. G.-C. And Tsao. T.-P.: Application of a Fuzzy Neural Network Combined With a Chaos Genetic Algorithm and Simulated Annealing to Short-Term Load Forecasting. IEEE Transaction On Evolutionary Comp. Vol. 10. (2006) 330-340
13. Eklund. N. H. W.: Using Genetic Algorithms to Estimate Confidence Intervals for Missing Spatial Data. IEEE Trans. On Systems. Man. And Cybernetics-Part C: Applications And Reviews Vol. 36. (2006) 519-524
14. G. T. Pulido. "On the Use of Self-Adaptation and Elitism for Multiobjective Particle Swarm Optimization". PhD Thesis in Electrical Eng. Department Computer Science Section. Centro de Investigación y Estudios Avanzados del Inst. Politécnico Nacional. (2005)

Automatic Design of ANNs by Means of GP for Data Mining Tasks: Iris Flower Classification Problem

Daniel Rivero, Juan Rabuñal, Julián Dorado, and Alejandro Pazos

Department of Information & Communications Technologies, Campus Elviña
15071, A Coruña, Spain
{drivero, julian, juanra, apazos}@udc.es
<http://sabia.tic.udc.es>

Abstract. This paper describes a new technique for automatically developing Artificial Neural Networks (ANNs) by means of an Evolutionary Computation (EC) tool, called Genetic Programming (GP). This paper also describes a practical application in the field of Data Mining. This application is the Iris flower classification problem. This problem has already been extensively studied with other techniques, and therefore this allows the comparison with other tools. Results show how this technique improves the results obtained with other techniques. Moreover, the obtained networks are simpler than the existing ones, with a lower number of hidden neurons and connections, and the additional advantage that there has been a discrimination of the input variables. As it is explained in the text, this variable discrimination gives new knowledge to the problem, since now it is possible to know which variables are important to achieve good results.

1 Introduction

ANNs are learning systems that have solved a large amount of complex problems related to different disciplines (classification, clustering, regression, etc.) [1]. The interesting characteristics of this powerful technique have induced its use by researchers in different environments [2].

Nevertheless, the use of ANNs has some problems mainly related to their development process. This process can be divided into two parts: architecture development and training and validation. As the network architecture is problem-dependant, the design process of this architecture used to be manually performed, meaning that the expert had to test different architectures and train them until finding the one that achieved best results after the training process. The manual nature of the described process determines its slow performance although the recent use of ANNs creation techniques have contributed to achieve a more automatic procedure.

The technique described in this paper allows the automatically obtaining of ANNs with no need of human participation. This technique is applied to a well-known problem: the Iris Flower classification Problem [3]. Results show how this technique can solve the problem. Moreover, it is capable of obtaining simpler networks than the existing ones for solving this problem.

2 State of the Art

2.1 Genetic Programming

Genetic Programming (GP) [4] is based on the evolution of a given population. In this population, every individual represents a solution for a problem that is intended to be solved. The evolution is achieved by means of selection of the best individuals – although the worst ones have also a little chance of being selected – and their mutual combination for creating new solutions. This process is developed using selection, crossover and mutation operators. After several generations, it is expected that the population might contain some good solutions for the problem.

The GP encoding for the solutions is tree-shaped, so the user must specify which are the terminals (leaves of the tree) and the functions (nodes capable of having descendants) for being used by the evolutionary algorithm in order to build complex expressions.

The wide application of GP to various environments and its consequent success are due to its capability for being adapted to numerous different problems. Although the main and more direct application is the generation of mathematical expressions [5], GP has been also used in others fields such as rule generation [6], filter design [7], etc.

2.2 ANN Development with EC Tools

The development of ANNs is a subject that has been extensively dealt with very diverse techniques. The world of evolutionary algorithms is no exception, and proof of that is the great amount of works that have been published about the different techniques in this area, even with GAs or GP [4] [13] [15] [19] [20] [21]. These techniques follow the general strategy of an evolutionary algorithm: an initial population consisting of different types of genotypes, each one of them codifying different parameters (typically, the weight of the connections and/or the architecture of the network and/or the rules of learning), and is randomly created. This population is evaluated in order to determine the goodness of each individual. Afterwards, this group is made to evolve repeatedly by means of distinct genetic operators (replication, crossover, mutation, etc.) until a determined termination criteria is fulfilled (for example, a sufficiently good individual is obtained, or that a predetermined maximum number of generations is achieved).

As a general rule, the field of ANNs generation using evolutionary algorithms is divided into three main fields: evolution of weights, architectures and learning rules.

First, the weight evolution starts from an ANN with an already determined topology. In this case, the problem to be solved is the training of the connection weights, attempting to minimize the network failure. Most of training algorithms, as back-propagation (BP) algorithm, are based on gradient minimization, which presents several inconveniences [8], mainly the possibility of getting stuck into a local minimum of the fitness function. With the use of an evolutionary algorithm, the weights can be represented either as the concatenation of binary values [9] or of real numbers [10]. The main disadvantage of this type of encoding is the permutation problem. This problem means that the order in which weights are taken at the vector might cause

that equivalent networks might correspond to completely different chromosomes, making the crossover operator inefficient.

Second, the evolution of architectures includes the generation of the topological structure. This means establishing the connectivity and the transfer function of each neuron. The network architecture is highly important for the successful application of the ANN, since the architecture has a very significant impact on the processing ability of the network. Therefore, the network design, traditionally performed by a human expert using trial and error techniques on a group of different architectures, is crucial. The automatic architecture design has been possible thanks to the use of evolutionary algorithms. In order to use them to develop ANN architectures, it is needed to choose how to encode the genotype of a given network for it used by the genetic operators.

At the first option, direct encoding, there is a one-to-one correspondence between every one of the genes and their subsequent phenotypes. The most typical encoding method consists of a matrix that represents an architecture where every element reveals the presence or absence of connection between two nodes [11]. These types of encoding are generally quite simple and easy to implement. However, they also have a large amount of inconveniences as scalability [12], the incapability of encoding repeated structures, or permutation [13].

Apart from direct encoding, there are some indirect encoding methods. In these methods, only some characteristics of the architecture are encoded in the chromosome. These methods have several types of representation. Firstly, the parametric representations represent the network as a group of parameters such as number of hidden layers, number of nodes for each layer, number of connections between two layers, etc [14]. Although the parametric representation can reduce the length of the chromosome, the evolutionary algorithm performs the search within a restricted area in the search space containing all the possible architectures. Another non direct representation type is based on grammatical rules [12]. In this system, the network is represented by a group of rules, shaped as production rules that make a matrix that represents the network, which has several restrictions.

The growing methods represent another type of encoding. In this case, the genotype does not encode a network directly, but it contains a group of instructions for building up the phenotype. The genotype decoding will consist on the execution of those instructions [15].

With regards to the evolution of the learning rule, there are several approaches [16], although most of them are only based on how learning can modify or guide the evolution and also on the relationship among the architecture and the connection weights.

2.3 Iris Flower Problem

The iris flower data were originally published by Fisher on 1936 as examples of discriminant analysis and cluster analysis [3]. Four parameters, sepal length, sepal width, petal length and petal width, were measured in millimeters on 50 iris specimens from each of three species, *Iris setosa*, *Iris versicolor* and *Iris virginica*. Given the four parameters, one should be able to determine which of the three classes a specimen belongs to. There are 150 data points listed in the database, taken from UCI [18]. As was already studied, with only two variables (petal length and petal width), a higher

discrimination for the three classes, a fitness of a 98% success using only these two variables, can be obtained (i.e., 147 correct classifications out of 150 data points) [22].

This problem has already been solved with different tools, ANNs between them. Martinez and Goddard [23] have proven that a maximum adjustment of 98.67 % correct answers (2 errors) is achieved with 6 neurons in the hidden layer. With the system put forward by Rabuñal [24] and 5 hidden neurons, tangent hyperbolic activation functions and threshold function of 0.5 in the output neurons, the previous register has improved (with regard to the number of hidden neurons), reaching also a 98.67% of correct answers. Results obtained with the technique described in this paper will show how smaller networks can be obtained in order to solve this problem.

3 Model

The GP-development of ANNs is performed by means of the GP typing property [17]. This property provides the ability of developing structures that follow a specific grammar. In this case, the nodes to be used, as described in previous works [25], are the following:

- ANN. Node that defines the network. It appears only at the root of the tree. It has the same number of descendants as the network expected outputs, each of them a neuron.
- n-Neuron. Node that identifies a neuron with n inputs. This node will have $2*n$ descendants. The first n descendants will be other neurons, either input or hidden ones. The second n descendants will be arithmetical sub-trees. These sub-trees represent real values. These values correspond to values of the respective connection weights of the input neurons – the first descendants – of this neuron.
- n-Input neuron. Nodes that define an input neuron which receives its activation value from the input variable n. These nodes will not have any descendants.
- Finally, the arithmetic operators set $\{+, -, *, \%, / \}$, where % designs the operation of protected division (returns 1 as a result if the divisor is 0). They will generate the values of connection weights (sub-trees of the n-Neuron nodes). These nodes perform operations among constants in order to obtain new values. As real values are also needed for such operations, they have to be introduced by means of the addition of random constants to the terminal set in the range $[-4, 4]$.

ANNs can be generated with these operator sets. However, these networks would not allow, for a given neuron, the existence of output connections to more than one different neuron. For such reason, the system has been endowed with a list where neurons are being added as the evaluation of the tree progresses, and an index that points at one specific element of the list. In order to extract neurons from the list, and therefore to operate with it, the operator sets is added with the following operators:

- “Forward”. This node advances the index list one unit. This node has one descendant.
- “Pop”. This node extracts from the list the neuron at the position pointed by the index. This node substitutes the evaluation of a neuron, as it gives back an already existing one, so it has no descendants.

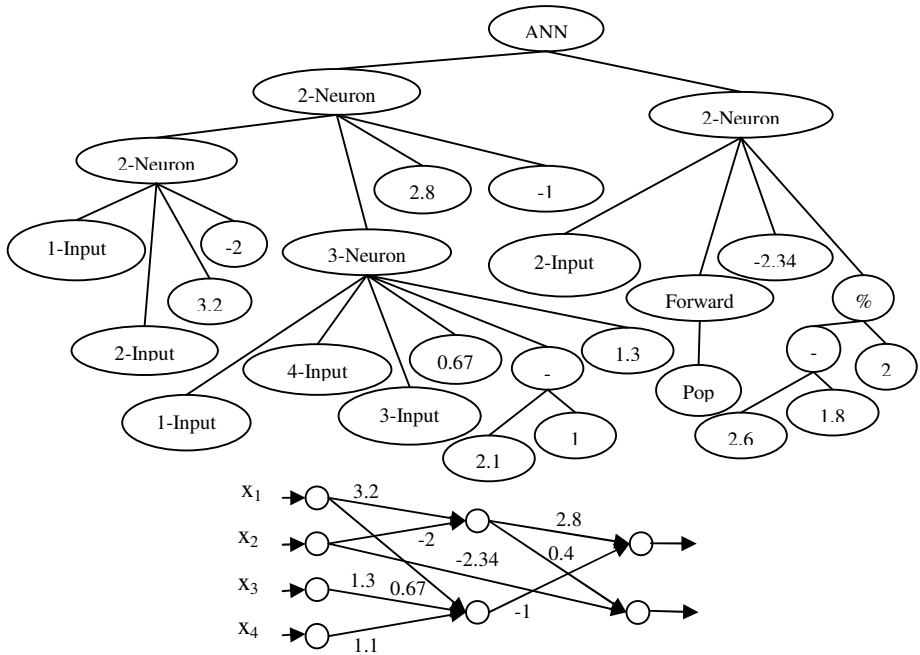


Fig. 1. GP tree and its resulting network

Every time a neuron is created, it is added to the list once its descendants have been evaluated. In such way, they are not allowed to reference that neuron, so recurrent links will be avoided.

Note that, during the neuron creation, a given neuron - either an input one or a referenced one - can be repeated several times as predecessor. In such case, there is no new input connection from that processing element, but the weight of the already existing connection will be added with the value of the new connection.

Fig. 1 shows an example of a GP that, using these nodes, represent an ANN.

Once the tree has been evaluated, the genotype turns into phenotype. In other words, it is converted into an ANN with its weights already set (thus it does not need to be trained) and therefore can be evaluated. The evolutionary process demands the assignation of a fitness value to every genotype. Such value is the result after the evaluation of the network with the pattern set representing the problem. This result is the mean square error (MSE) of this evaluation.

Nevertheless, this error value considered as fitness value has been modified in order to induce the system to generate simple networks. The modification has been made by adding a penalization value multiplied by the number of neurons of the network. In such way, and given that the evolutionary system has been designed in order to minimize an error value, when adding a fitness value, a larger network would have a worse fitness value. Therefore, the existence of simple networks would be preferred as the penalization value that is added is proportional to the number of neurons at the ANN. The calculus of the final fitness will be as follows:

$$fitness = MSE + N * P$$

Where MSE is the mean square error of the ANN within the group of training patterns, N is the number of neurons of the network and P is the penalization value for such number.

4 Results

The system described here has been applied to solve this particular problem. After doing the normalization of the four variables between 0 and 1, the whole database has been applied to the system. It was run with the following configuration parameters, which have shown to give good results on previous works [25]:

- Crossover rate: 95%.
- Mutation probability: 4%.
- Selection algorithm: 2-individual tournament.
- Creation algorithm: Ramped Half&Half.
- Tree maximum height: 5.
- Maximum inputs for each neuron: 6.
- Penalization: 0.00001.

Different networks were obtained that solved this problem. A small comparison of the networks found can be seen on Table 1.

Table 1. Comparison of the architectures found

| Method | Hits | Accuracy | Hidden | Connections | Figure |
|----------|------|----------|--------|-------------|---------------|
| [24] | 148 | 98.66 % | 5 | 35 | - |
| | 149 | 99.33 % | 3 | 15 | Fig. 2 b) |
| Proposed | 148 | 98.66 % | 1 | 11 | Fig. 3 a), b) |
| here | 147 | 98 % | 1 | 9 | Fig. 4 a) |
| | 146 | 97.33 % | 1 | 10 | Fig. 5 b) |
| | 145 | 96.66 % | 1 | 10 | Fig. 6 a) |

As can be seen on this table, the architectures found are very simple, with a small number of hidden nodes and connections, improving the results found in the previous works. Even the accuracy was improved, reaching a 98.66% success (149 correct answers).

Fig. 2 shows two networks with this accuracy (99.33%). Network b) is the one shown on Table 1, with 3 hidden neurons and 15 connections. Network a) is a little more complicated, with the same number of neurons (3) but more connections (19). However, this latter network has another important feature: input X_1 (sepal length) was not used. So, it can be concluded that this input is not useful for the correct classification of 149 out of the 150 data points.

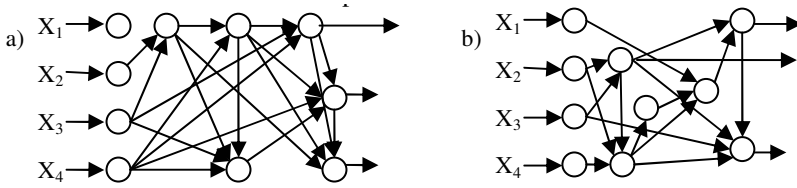


Fig. 2. 99.33% accuracy (1 fail) ANNs

Fig. 3 shows two other networks that solve the problem with an accuracy of 98.66%, i.e., it fails on two data points. Both networks show that the variable X_1 (sepal length) was not needed to do this classification. These networks are much simpler than the ones found using other systems [23] [24] and gave the same accuracy.



Fig. 3. 98.66% accuracy (2 fails) ANNs

Fig. 4 shows three different ANNs that give an accuracy of 98%. As was explained, this accuracy was proved to be able to be found with only two inputs [22]. This is also shown on this figure, since network c) does not need neither X_1 nor X_2 (sepal length and width) to perform the 98% correct classifications. Fig. 4 a) and b) show small networks found that perform this accuracy too.

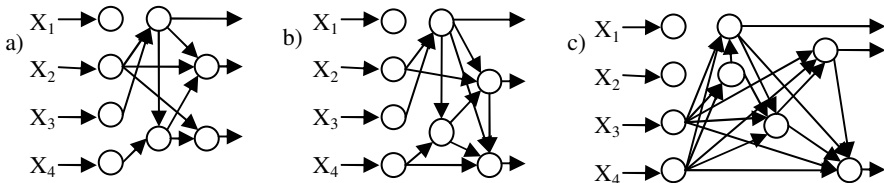


Fig. 4. 98% accuracy (3 fails) ANNs

Some networks with an accuracy of 97.33% are shown on Fig. 5. On two of them, neither X_1 nor X_2 were used to make the classification.

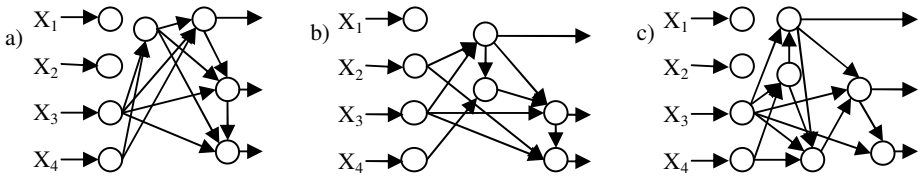


Fig. 5. 97.33% accuracy (4 fails) ANNs

Finally, Figure 6 shows three different networks that perform the classification with an accuracy of 96.66%, i.e., it fails on 5 data points. The interesting of the networks b) and c) are that they don't use variables X_1 (sepal length) and X_3 (petal length) to perform this classification. As was previously explained, variables X_3 and X_4 (petal length and width) are needed to correctly classify the 98% of the data points. However, as can be seen on these networks, variable X_3 was found not to be needed to classify the 96.66% of the data.

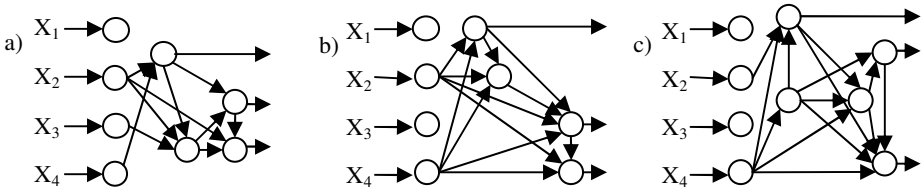


Fig. 6. 96.66% accuracy (5 fails) ANNs

5 Conclusions

As the results section shows, the technique described in this paper can develop ANNs which provide good results at solving this particular problem. This problem is only an example of an application of this technique to a Data Mining problem, but is well documented, so it can be deeply analyzed and its results compared to other Data Mining tools. Results show that the method described here provides results better than many other knowledge extraction methods, and better than all of the remaining ANN development methods.

But this system has other different advantages: First, the developed ANNs are much simpler than the existing ones for solving this problem (many ANNs were obtained with only one hidden neuron), and they provide better results; and second, a discrimination between the variables has been found, i.e., many ANNs have not found some variables to be useful for achieving a particular accuracy in solving this problem.

This latter advantage is very important, since it gives new knowledge to this environment. In particular, from the obtained results, it can be concluded:

- To correctly classify 149 out of the 150 data point, the variable sepal length is not necessary.
- The variable petal length, which is necessary for achieving a correct classification higher than 145 correct classifications, is not necessary (together with the variable sepal length, which is neither necessary) for correctly classify 145 of the 150 data points.
- As was already known and found here again, both variables sepal length and sepal width are not necessary for the correct classification of 147 data points.

Therefore, it can be concluded that the most important variable for the classification is petal width, which is the only that is needed in all of the cases. On the other side, the variable sepal length has very few influence, since it was not found to be used by the majority of the networks.

Acknowledgments. This work was supported in part by the Spanish Ministry of Education and Culture (Ref. TIC2003-07593, TIN2006-13274), the IMBIOMED network (Ref P10/52048) financed by the Carlos III Health Institute, grants from the General Directorate of Research of the Xunta de Galicia (Ref. PGIDIT03-PXIC10504PN PGIDIT04-PXIC10503PN, PGIDIT04-PXIC10504PN), and the European project INTERREG (Ref. IIIA-PROLIT-SP1E194/03).

The development of the experiments described in this work, has been performed with equipments belonging to the Super Computation Center of Galicia (CESGA).

References

1. Haykin, S., *Neural Networks* (2nd ed.), Englewood Cliffs, NJ: Prentice Hall (1999)
2. Rabuñal, J.R., Dorado J., (eds.) *Artificial Neural Networks in Real-Life Applications*, Idea Group Inc (2005)
3. Fisher, R. A. The use of multiple measurements in taxonomic problems. *Annals of Eugenics* (1936) 179-188
4. Koza, J. R., *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, Cambridge, MA, MIT Press (1992)
5. Rivero D., Rabuñal J.R., Dorado J., Pazos A., *Time Series Forecast with Anticipation using Genetic Programming*, IWANN 2005 (2005) 968-975
6. Bot, M., *Application of Genetic Programming to Induction of Linear Classification Trees*, Final Term Project Report, Vrije Universiteit, Amsterdam (1999)
7. Rabuñal J.R., Dorado J., Puertas J., Pazos A., Santos A., Rivero D., *Prediction and Modeling of the Rainfall-Runoff Transformation of a Typical Urban Basin using ANN and GP*, *Applied Artificial Intelligence* (2003)
8. Sutton, R.S., Two problems with backpropagation and other steepest-descent learning procedure for networks, *Proc. 8th Annual Conf. Cognitive Science Society*, Hillsdale, NJ: Erlbaum (1986) 823-831
9. Janson D.J., Frenzel J.F., *Training product unit neural networks with genetic algorithms*, *IEEE Expert*, Vol. 8 (1993) 26-33
10. Greenwood G.W. *Training partially recurrent neural networks using evolutionary strategies*, *IEEE Trans. Speech Audio Processing*, Vol. 5 (1997) 192-194

11. Alba E., Aldana J.F., Troya J.M., Fully automatic ANN design: A genetic approach, Proc. Int. Workshop Artificial Neural Networks (IWANN'93), Lecture Notes in Computer Science, vol. 686. Berlin, Germany: Springer-Verlag (1993) 399-404
12. Kitano H., Designing neural networks using genetic algorithms with graph generation system, Complex Systems, Vol. 4 (1990) 461-476
13. Yao X., Liu Y., Towards designing artificial neural networks by evolution, Appl. Math. Computation, Vol. 91, No. 1 (1998) 83-90
14. Harp S.A., Samad T., Guha A., Toward the genetic synthesis of neural networks, Proc. 3rd Int. Conf. Genetic Algorithms and Their Applications, J.D. Schafer, Ed. San Mateo, CA: Morgan Kaufmann (1989) 360-369
15. Nolfi S. & Parisi D., Evolution of Artificial Neural Networks, Handbook of brain theory and neural networks, Second Edition, Cambridge, MA: MIT Press (2002) 418-421
16. Turney P. Whitley D., Anderson R., Special issue on the baldwinian effect, Evolutionary Computation, Vol. 4, No. 3 (1996) 213-329
17. Montana D.J., Strongly typed genetic programming, Evolutionary Computation, Vol. 3, No. 2 (1995) 199-200
18. Mertz C.J., Murphy P.M., UCI repository of machine learning databases. <http://www-old.ics.uci.edu/pub/machine-learning-databases> (2002)
19. Cantú-Paz E., Kamath C., An Empirical Comparison of Combinations of Evolutionary Algorithms and Neural Networks for Classification Problems, IEEE Transactions on systems, Man and Cybernetics – Part B: Cybernetics (2005) 915-927
20. Herrera F., Hervás C., Otero J., Sánchez L., Un estudio empírico preliminar sobre los tests estadísticos más habituales en el aprendizaje automático, R. Giraldez, J.C. Riquelme, J.S. Aguilar (Eds.) Tendencias de la Minería de Datos en España, Red Española de Minería de Datos y Aprendizaje (2004) 403-412
21. Gruau F., Genetic Micro Programming of Neural Networks, Advances in Genetic Programming, K. Kinnear, Ed., Cambridge, MA:MIT Press (1994) 495-518
22. Duch, W., Adamczak, R., Grabczewski, K., A new methodology of extraction, optimisation and application of crisp and fuzzy logical rules. IEEE Transactions on Neural Networks, Vol. 11, No. 2 (2000)
23. Martinez, A., Goddard, J., Definición de una red neuronal para clasificación por medio de un programa evolutivo. Mexican Journal of Biomedical Engineering, Vol. 22 (2001) 4-11
24. Rabuñal, J. R. Entrenamiento de redes de neuronas artificiales mediante algoritmos genéticos. University of A Coruña, Spain. Graduate Thesis (1999)
25. Rivero D., Dorado J., Rabuñal J., Pazos A., Using Genetic Programming for Artificial Neural Network Development and Simplification, Proceedings of the 5th WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics (CIMMACS'06), WSEAS Press, (2006) 65-71

FPGA Implementation of Evolvable Characters Recognizer with Self-adaptive Mutation Rates

Jin Wang¹, Chang Hao Piao², and Chong Ho Lee¹

¹ Department of Information & Communication Engineering,
Inha University, Incheon, Korea
wangjin_liips@yahoo.com.cn

² Department of Automation Engineering, ChongQing University of Posts and
Telecommunications, Chongqing, China

Abstract. As an alternative to traditional artificial neural network approaches to pattern recognition, a hardware-implemented evolvable characters recognizer is presented in this paper. The main feature of the proposed evolvable system is that all the components including the evolutionary algorithm (EA), fitness calculation, and virtual reconfigurable circuit are implemented in a Xilinx Virtex xcv2000E FPGA. This allows for a completely pipelined hardware implementation and yields a significant speedup in the system evolution. In order to optimize the performance of the evolutionary algorithm and release the users from the time-consuming process of mutation parameters tuning, a self-adaptive mutation rate control scheme is also introduced. An analysis of experimental results demonstrates that the proposed evolvable system using self-adaptive mutation rates is superior to traditional fixed mutation rate-based approaches.

1 Introduction

In recent years, evolvable hardware (EHW) has been employed in the high-speed pattern recognition system design automation field, as an alternative to conventional artificial neural networks (ANN) approaches [1]. The significant benefits of the EHW approach to pattern recognition compared with ANN are the high processing speed of the hardware and the readability of the learned result, which have been discussed in literature [2]. Some EHW-based pattern recognition systems have previously been proposed. Torresen [3] introduced an incremental evolution-based EHW approach for the evolution of 16 characters recognition system. In report [4], another kind of evolvable recognition system was proposed by Torresen for recognizing speed limit sign numbers. Both of them are based on the extrinsic evolution strategy in which fitness evaluation of each candidate circuit is simulated by software. Several studies [5, 6] have proved that the evolutionary speed of the extrinsic EHW is much slower than intrinsic evolutions in which each evolved candidate circuit is built and tested on hardware. In the domain of evolvable pattern recognition system, an intrinsic evolution approach with Xilinx 4025 FPGA was proposed in [2]. In this report, Iwata et al directly mapped the configuration bit streams of the FPGA as the chromosome.

With the idea of realizing incremental evolution by virtual reconfigurable circuit-based intrinsic EHW, we have proposed different FPGA-implemented evolvable characters classification systems as the scalability approaches to EHW in literatures [7, 8].

In this paper, a novel self-adaptive mutation rate-based intrinsic EHW is introduced to evolve a 16 characters recognition system. In order to allow the EA to optimize its performance in the running period and to eliminate the non-trivial EA parameter tuning time, a self-adaptive mutation rate control scheme is implemented in our experiments. A Xilinx Virtex xcv2000E FPGA fitted in a Celoxica RC1000 PCI board [9] is utilized as our hardware experimental platform. The work presented herein mainly focuses on demonstrating the possibility and efficiency of the FPGA implementation of evolvable characters recognizer and the benefits of utilizing a self-adaptive mutation rate control scheme, rather than evolving a complicated characters recognition system which have been successfully implemented in our previous works [7, 8] with different incremental evolution strategies.

The remainder of this paper is organized as follows. Section 2 describes the basic concept of the proposed evolvable characters recognizer. Hardware realization of the self-adaptation mutation rate-based intrinsic evolvable system is introduced in section 3. Section 4 presents experimental results. Section 5 discusses the obtained results and the final section provides conclusions.

2 Evolvable Characters Recognizer

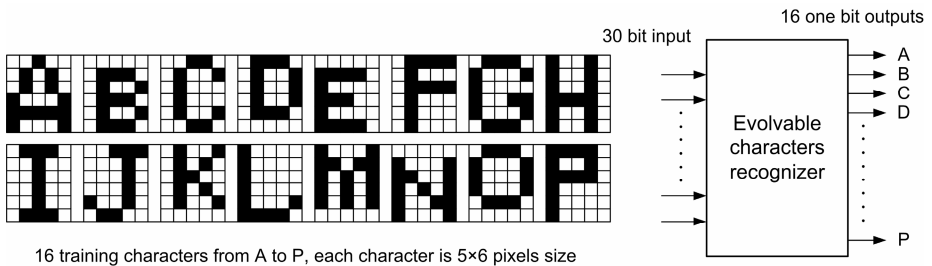


Fig. 1. Diagram of characters recognizer for patterns A to P identification

Characters recognizer can be evolved from both the gate and function level [1]. This paper concentrates on a gate level evolution. The target evolvable system is expected to identify 16 different binary patterns (characters from A to P) of 5×6 pixels, where each pixel can be 0 or 1. The characters recognition procedure consists of two phases. The first phase is the characters recognizer training phase. In this phase, proposed evolvable characters recognizer works as an adaptive machine that tries to minimize the diversity between the desired output and the actual system output by using EA learning. Characters from A to P are employed as a training patterns set (see Fig. 1). In each system clock, one character is processed by the evolvable system as a training vector. To process each input character of 5×6 pixels, the proposed character recognizer includes a 30-bit input port. Each 1 bit input responds to one pixel in the

input characters. As shown in Fig. 1, each single bit output port of the evolvable characters recognizer corresponds to one character in the training set. Therefore, the proposed system consists of 16 single bit output ports. The second phase of the character recognition procedure is the character distinguishing process. During this process, the output port of the evolved system corresponding to the input character should be 1; synchronously the other 15 single bit outputs should be 0. According to this feature, 16 different characters in the training set can be recognized individually.

3 Hardware Realization

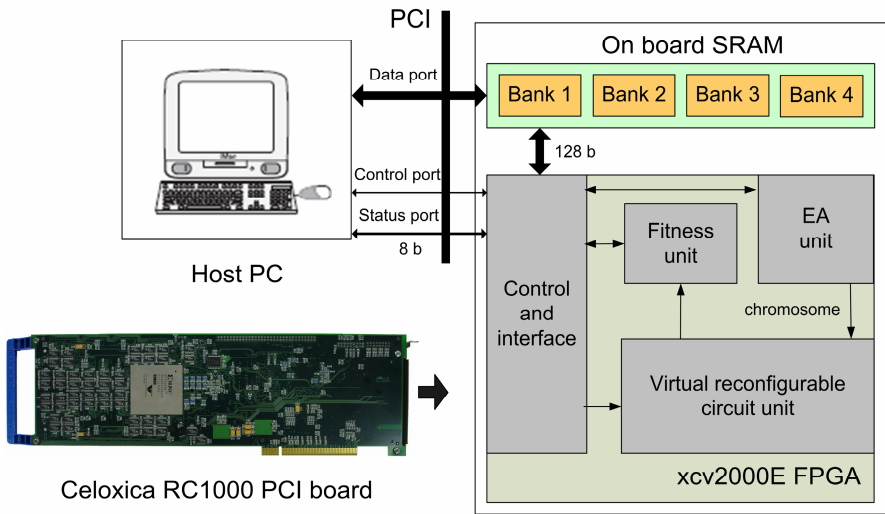


Fig. 2. Block diagram of complete FPGA-implemented evolvable system

In order to evolve the proposed character recognizer, a virtual reconfigurable circuit architecture-based intrinsic evolvable system is employed as our experimental platform. This architecture was inspired by the Cartesian Genetic Programming (CGP) [10] and was first proposed by Sekanina in literature [11] as a feasible and efficient approach to realizing intrinsic evolvable image operators on a commercial FPGA.

The main feature of our proposed evolvable system is that all the components of the system including a virtual reconfigurable circuit (VRC) unit, EA unit, fitness unit, and control and interface are realized in a Xilinx Virtex xcv2000E FPGA that is fitted in a Celoxica RC1000 PCI board. The block structure of our evolvable system is shown in Fig. 2. In the system evolution process, all system evolutionary operations are executed on FPGA. The function of the host PC is only for communicating with the RC1000 PCI board, i.e. for writing/reading data to/from on board SRAM and sending/receiving the start/stop signals to/from the FPGA by the 8 bits status port and the 1 bit control port. In the FPGA, control and interface manages the operations of other components of EHW by using a finite state machine and communicates with the

host PC and onboard memory. The EA unit implements the evolutionary operations and generates configuration bits to configure the VRC unit. System function and evolution are performed in the VRC unit. The fitness unit calculates individual fitness value by comparing the outputs from the VRC unit with predefined expected system outputs.

In the evolution process of EHW, there are several EA parameters that need to be predefined. This specifies some aspects of how the EA search will be conducted, e.g. the population size, the mutation rate, etc. This paper focuses on discussion of the mutation operator, as it has been suggested to be the most sensitive EA control parameter [12, 13]. A general way of setting the mutation parameter is parameter tuning. In order to find an optimized constant mutation rate for a certain application, researchers generally base their choices on tuning the mutation parameters by hand, that is, experimenting with different values and selecting one that exhibits the best performance. However, this process is very time-consuming and the tuned result is only efficient for some limited EA settings. On the other hand, EA learning is an intrinsically dynamic, adaptive process. Generally, different values of mutation rate are desired at different stages of the evolutionary process in order to achieve balance between global and local searches. The use of a fixed mutation rate seems to be in conflict with this requirement of EA. Unfortunately, most reported CGP-inspired evolvable hardware [3, 6, 7, 8, 14, 15] still employs a rigid mutation rate, as most of reported adaptive mutation rate control schemes [12] are complicated and unfeasible for hardware.

In this paper, we propose a novel and hardware feasible self-adaptive mutation rate control scheme for our FPGA-implemented EHW. Self-adaptation is based on the concept of the evolution of the mutation parameters, which adjust mutation rates to adapt the problem during EA running. The parameters that control the mutation rates of the chromosomes are encoded into their corresponding chromosomes as additional genes and also undergo evolutionary operations (selections and mutations). The motive of this scheme is that better values of these encoded mutation parameters will lead to better individuals, which in turn are more likely to survive and produce offspring and hence propagate better parameter values. The important feature of self-adaptation of the mutation rate is that potentially the algorithm can be adjusted to the problem while solving that problem.

3.1 Virtual Reconfigurable Circuit Unit

The VRC unit includes 30 bits input and 16 single bit outputs. Traditional CGP scheme [10] employed a string of integers as chromosome to represent the functions and connections of a grid of 3-inputs nodes. However, in our proposed virtual reconfigurable circuit-based EHW, for the hardware implementation, the system genotype is revised as a linear binary bits string that illuminates the connections and functions of a 2-inputs function elements (FE) array. The system phenotype is presented by an interconnected network of enabled FEs which is created in the FEs array by system genotype mapping. To self-adapt the mutation rate, rather than fix it, the mutation rate control parameters are also encoded into the chromosome as additional genes. The genotype and the mapping process of the genotype to phenotype

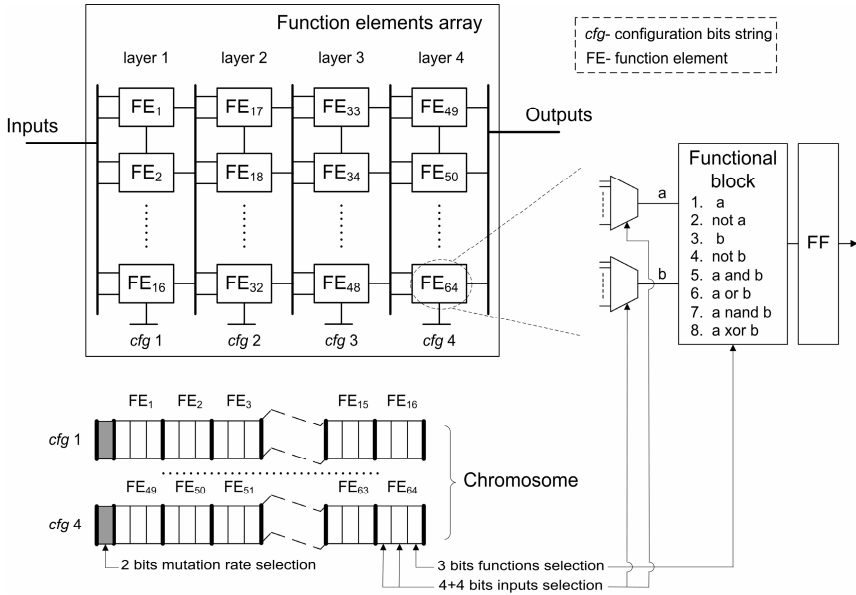


Fig. 3. Genotype-phenotype mapping

are illustrated in Fig. 3, in which the additional genes for self-adaptive mutation rates selection are marked as gray blocks.

For our proposed application, the FEs array consists of 4 FE layers. 16 uniform FEs are placed in each layer. Each FE's two inputs in layer l ($l=2, 3, 4$) can be connected to any one output of FEs in layer $l-1$. As shown in Fig.3, the input connections of each FE are selected by its two equipped 16-to-1 multiplexers. In layer 1, each FE employs two 32-to-1 multiplexers. Therefore, each input of FE in layer 1 can be connected to any one of the 30 bits system inputs or defined as a bias of value 1 or 0. In the FEs array, each FE in the last layer corresponds to one system output. This means the circuit output connection is not evolvable. Each FE in layers 2,3,4 can have one of eight functions, as is evident from Fig. 3. Only two functions of buffer a and inverter b are available for FEs in layer 1. Each FE is equipped with a Flip-flop to support pipeline processing. A layer of FEs is considered as a single stage of the pipeline. 16 FEs in the same layer can be configured simultaneously and we need 4 system clocks to completely change the configuration of the FEs array. According to this feature of the reconfiguration of the FEs array, each chromosome is divided into 4 individual configuration bits strings (cfg). Each cfg undergoes independent evolutionary operations and is responsible for the configuration of its corresponding single layer of FEs (as shown in Fig. 3, the intact chromosome is divided into 4 parts: $cfg1$, $cfg2$ etc.). Each FE needs 11 bits (5+5+1 bits in layer 1, 4+4+3 bits in the other layers) to determine its input connections and functions. On the other hand, each cfg needs additional 2 bits (it is configurable) to select an enabled mutation rate from four candidates. In total, the chromosome length is $4 \times (11 \times 16 + 2) = 712$ bits. Chromosome is uploaded from EA unit. By continuously altering the chromosome, the interconnections and functions of FEs array can be evolved.

3.2 Evolutionary Algorithm Unit

The evolutionary algorithm employed in the EA unit is according to the $1 + \lambda$ evolutionary strategy, where $\lambda = 4$. In our experiment, evolution is only based on the mutation and selection operators; crossover is not taken into account. Because the FEs array has to be configured layer by layer, the chromosome is divided into 4 individual configuration bits strings which undergo mutation operations individually. Two mutation operators are performed in the proposed EA: (1) to the configuration bits strings of the FEs array; and (2) to the additional genes that decide the mutation rates of their corresponding configuration bits strings. Each additional genes controls the mutation probability p_m of its corresponding *cfg* according to some predefined descriptions. For example, we choose the mutation rates 0.2%, 0.4%, 0.8%, and 1.6% as candidate mutation probabilities. The size of additional genes is 2 bits. Active p_m can be selected based on the following rule:

| Additional 2 bits genes | 00 | 01 | 10 | 11 |
|-------------------------|------|------|------|------|
| p_m | 1.6% | 0.8% | 0.4% | 0.2% |

The size of the additional genes and the value of mutation probability p_m are configurable.

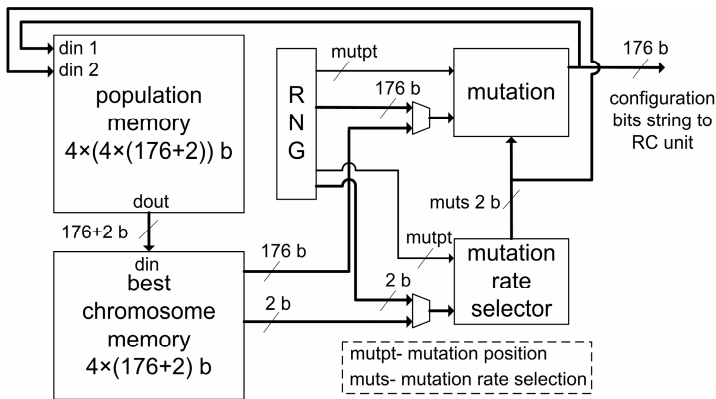


Fig. 4. Implementation of self-adaptive mutation rate control in evolutionary algorithm unit

Fig. 4 shows the hardware realization of the EA unit. The EA unit consists of a population memory, a best chromosome memory, a random number generator (RNG), a mutation unit, and a mutation rate selector. When the EA unit receives the start signal from control and interface, the population memory is filled by four chromosomes, which are the mutated versions of four $4 \times (176+2)$ bits random numbers generated by the RNG with cellular automata [16]. In the mutation process, the $4 \times (176+2)$ bits number is divided into two parts: 4×176 bits and 4×2 bits numbers, which are processed by the mutation unit and mutation rate selector, respectively. The mutation unit processes each 4×176 bits data in 4 clocks per 176 bits. Synchronously, 4×2 bits additional genes are also mutated in the mutation rate

selector in 4 clocks per 2 bits with a constant mutation rate of 12.5%. The mutation rate is defined as the percentage of the bits of the entire 4×2 bits additional genes, which will undergo mutation. The mutated version of four 2 bits additional genes will decide the mutation probabilities of their corresponding four 176 bits configuration bits strings, respectively. The fitness value of each individual will be evaluated by using the four mutated 176 bits strings to configure the virtual reconfigurable circuit unit. This means that the encoded additional genes of 4×2 bits affect only the generation of offspring, and have no direct impact on the fitness evaluation of an individual. After all chromosomes of the initial population have been evaluated, the best one is selected and saved in the best chromosome memory as the parent chromosome. The new population is generated using the parent chromosome and its 4 mutants. If the fitness value of a mutant is higher than the parent chromosome, then the mutated chromosome replaces the parent in the best chromosome memory. This is repeated until the stop criteria of EA are satisfied. The stop criteria of EA are defined as: (1) EA finds the expected solution; (2) The predefined generation number (2^{30}) is exhausted.

3.3 Fitness Unit

In the proposed system, system training set includes 16 test vectors (16 characters from A to P). The fitness unit evaluates the circuits uploaded to the virtual reconfigurable circuit unit by reading its output vectors and comparing them against the expected output vectors. The fitness function is calculated as follows:

$$Fitness = \sum_{vector} \sum_{output} x ; \quad \text{where } x = \begin{cases} 1 & \text{output} = \text{expect} \\ 0 & \text{output} \neq \text{expect} \end{cases} \quad (1)$$

In an output *vector*, each single bit *output* is compared with its corresponding expected system output (which is labeled as *expect*). If they are equal, the variable x will be presented as 1 and be added to the fitness function. *Fitness* is the sum values (the maximum fitness value is $16 \times 16 = 256$) for the compared results of all 16 single bit outputs (*output*) in the processed 16 output vectors (*vector*) set.

4 Experimental Results

The evolvable character recognizer was designed using VHDL and synthesized into a Xilinx Virtex xcv2000E FPGA using Xilinx ISE 6.3. With the diversity of the mutation rate control schemes, the device costs were slightly different, as shown in Table 1. According to the system synthesis report, the proposed systems can be operated at more than 90MHz. However, the actual hardware experiment was run at 30MHz for easier synchronization with the PCI interface.

As the pipeline process is supported by the proposed evolvable system, all EA operations time as well as reconfiguration time of FEs array could be overlapped by the process of fitness evaluation. Therefore, if the number of generations is $ngen$, the population size is p , the size of characters training set is S , and the hardware platform operates at f_m MHz, it is possible to express the system evolution time t as:

$$t = t_{init} + \frac{ngen \times p \times S}{f_m} \quad (2)$$

Where t_{init} is the time needed to generate the first output in pipeline process (several FPGA clocks only).

Table 1. Results for evolving a 16 characters recognizer by using different mutation control schemes

| Mutation scheme | Size of additional genes | Mutation Rate (%) | Total evolution time (avg.) | Number of generations | | Device cost (slices) |
|------------------------|--------------------------|-------------------|-----------------------------|-----------------------|-----------|----------------------|
| | | | | avg. | std. dev. | |
| Adapt. mutation rate | 4×2 bits | 1.6~0.2 | 1.18 sec | 552619 | 254594 | 6047 |
| | 4×2 bits | 0.8~0.1 | 0.69 sec | 325180 | 181018 | 5791 |
| | 4×3 bits | 0.8~0.1 | 0.77 sec | 362291 | 189634 | 6302 |
| Constant mutation rate | 0 bits | 0.1 | 1.77 sec | 829290 | 1017057 | 5616 |
| | | 0.2 | 1.12 sec | 525465 | 464265 | 5603 |
| | | 0.4 | 4.67 sec | 2189462 | 2429495 | 5614 |
| | | 0.8 | 4.78 sec | 2238728 | 1513587 | 5615 |
| | | 1.6 | 115.88 sec | 54318945 | 35729839 | 5841 |

The target 16 characters recognition systems have been evolved using different self-adaptive mutation and fixed mutation rate-based approaches, respectively. For the EAs with self-adaptive mutation rates, three different self-adaptive mutation control settings were tested in our experiments: (1) additional genes = 4×2 bits, $p_m=1.6\%$, 0.8%, 0.4%, 0.2%, (2) additional genes = 4×2 bits, $p_m=0.8\%$, 0.4%, 0.2%, 0.1%, and (3) additional genes = 4×3 bits, $p_m=0.8\%$, 0.7%, 0.6%, 0.5%, 0.4%, 0.3%, 0.2%, 0.1%. For comparison, traditional EAs with different fixed mutation probabilities were also executed in our experiments. We performed 50 runs per mutation rate setting and the initial seed for the initial population in each EA run was generated at random. All 50 EA runs were successful in all cases. One successful run means EA can find a feasible result circuit within predefined number of generations. Table 1 summarizes the experiments. All the average and stand deviation (std. dev.) results in Table 1 were calculated for 50 EA runs.

5 Discussion

In this paper, a complete FPGA implemented evolvable character recognizer is introduced. The proposed system could evolve the target 16 characters recognizer from scratch in relatively short time (no more than 1 second when a proper self-adaptive mutation rate control scheme was selected). When compared to the same algorithm described in C language running on an AMD Athlon64 3200+ CPU, our hardware implementation showed a performance increase of at least two orders of magnitude.

Different mutation rate settings have been tested in the experiments of evolving 16 characters recognizers. From Table 1, it can be observed that the performance of

the target evolvable system is very sensitive to the choice of the mutation probability. Under a rigid mutation rate, once outside the interval $p_m \in [0.1\%, 0.2\%]$, the number of generations required to reach the maximal fitness value (256) increases rapidly. For example, more than 103 times as many average numbers of generations are required as the fixed mutation probability is changed from 0.2% to 1.6%. On the other hand, for the proposed self-adaptive mutation, the performances of EAs are not very sensitive to the selection of the mutation probabilities interval. Three different partitions of self-adaptive mutation probabilities intervals have been tested in our experiments. All of them achieved the expected results under the close average number of generations. Another important observation listed in Table 1 is that the performance of the EA with a self-adaptive mutation operator outperforms all of the fixed mutation rates when the self-adaptive mutation rates interval is selected properly (e.g. under the mutation probabilities interval $p_m \in [0.1\%, 0.8\%]$, self-adaptive mutation scheme with 4×2 bits additional genes obtains a performance speedup of 1.6 times when compared to the selected best constant mutation probability, i.e. 0.2%).

6 Conclusions

In order to achieve the ability of high-speed pattern recognition, this paper has presented a complete FPGA implemented evolvable 16 characters recognition system. The proposed intrinsic evolvable system can evolve the target circuit in less than 1 second, significantly outperforming any reported conventional artificial neural networks or extrinsic EHW-based approaches. A hardware feasible self-adaptive mutation rate control scheme has been introduced for improving the performance of the evolvable hardware. To investigate the influence of the self-adaptive mutation rate control, its evolutionary results are compared with different fixed mutation rate-based evolutions. It is found that the performance of the proposed evolvable system using a self-adaptive mutation rates is remarkably superior to that of the traditional fixed mutation rate-based approaches. The proposed approach holds promise as a means of avoiding the time-consuming process of mutation parameter tuning.

Acknowledgments. This work was supported by INHA University Research Grant.

References

1. Yao, X., Higuchi, T.: Promises and Challenges of Evolvable Hardware. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, Vol. 29, No. 1 (1999) 87–97
2. Iwata, M. et al.: A Pattern Recognition System Using Evolvable Hardware. In: *Proc. of Parallel Problem Solving from Nature IV*, LNCS 1141, Springer-Verlag (1996) 761–770
3. Torresen, J.: A scalable approach to evolvable hardware. *Journal of Genetic Programming and Evolvable Machines*, Vol. 3, No. 3 (2002) 259–282
4. Torresen, J., Bakke, J.W., and Sekanina, L.: Recognizing Speed Limit Sign Numbers by Evolvable Hardware. In *Proc. of Parallel Problem Solving from Nature VIII*, Birmingham, UK, LNCS 3242, Springer-Verlag (2004) 682–691

5. Zhang, Y. et al.: Digital Circuit Design Using Intrinsic Evolvable Hardware. In: Proc. of the 2004 NASA/DoD Conference on the evolvable Hardware, IEEE Computer Society (2004) 55–63
6. Martínek, T., Sekanina, L.: An Evolvable Image Filter: Experimental Evaluation of a Complete Hardware Implementation in FPGA. In: Proc. of the 6th Int. Conference on Evolvable Systems: From Biology to Hardware, LNCS 3637, Springer-Verlag (2005) 76–85
7. Wang, J. et al.: Using Reconfigurable Architecture-Based Intrinsic Incremental Evolution to Evolve a Character Classification System. In: Proc. of the 2005 International Conference on Computational Intelligence and Security, CIS 2005, Part I, LNAI 3801, Springer-Verlag (2005) 216–223
8. Wang, J., Chong Ho Lee.: Introducing Partitioning Training Set Strategy to Intrinsic Incremental Evolution. In: Proc. of the 2006 Mexican International Conference on Artificial Intelligence, MICAI 2006, LNAI 4293, Springer-Verlag (2006) 272–282
9. Celoxica Inc., RC1000 Hardware Reference Manual V2.3, 2001
10. Miller, J.F., Thomson, P.: Cartesian Genetic Programming. In: Proc. of the Third European Conference on Genetic Programming, LNCS 1802, Springer-Verlag (2000) 121–132
11. Sekanina, L.: Virtual Reconfigurable Circuits for Real-World Applications of Evolvable Hardware. In: Proc. of the 5th International Conference Evolvable Systems: From Biology to Hardware, ICES 2003, LNCS 2606, Springer-Verlag (2003) 186–197
12. Rogen, D. et al.: Parameter Control in Evolutionary Algorithm. IEEE Transactions on Evolutionary Computation, Vol. 3, No. 2 (1999) 124–141
13. Stomeo, E. et al.: Mutation Rate for Evolvable Hardware. Transactions on Engineering, Computing and Technology, Vol. 7 (2005) 117–124
14. Sekanina, L., Friedl, S.: On Routine Implementation of Virtual Evolvable Devices Using COMBO6. In: Proc. of the 2004 NASA/DoD Conference on Evolvable Hardware, IEEE Computer Society (2004) 63–70
15. Kalganova, T.: Bidirectional Incremental Evolution in Extrinsic Evolvable Hardware. In: Proc. of the 2nd NASA/DoD Workshop on Evolvable Hardware, IEEE Computer Society (2000) 65–74
16. Wolfram, S.: Universality and Complexity in Cellular Automata. *Physica* (1984)10D:1–35

A Multi-gene-Feature-Based Genetic Algorithm for Prediction of Operon

Shuqin Wang^{1,2}, Yan Wang¹, Wei Du¹, Fangxun Sun¹, Xiumei Wang¹,
Yanchun Liang^{1,*}, and Chunguang Zhou¹

¹ College of Computer Science and Technology, Jilin University, Key Laboratory of Symbol Computation and Knowledge Engineering of the Ministry of Education, Changchun 130012, China

² School of Mathematics & Statistics, Northeast Normal University, Changchun, 130024, China
ycliang@jlu.edu.cn, wangsq562@nenu.edu.cn

Abstract. The prediction of operons is critical to reconstruction of regulatory networks at the whole genome level. In this paper, a multi-approach guided genetic algorithm is developed to prediction of operon. The fitness function is created by using intergenic distance of local entropy-minimization, participation of the same metabolic pathway, log-likelihood of COG gene functions and correlation coefficient of microarray expression data, which have been used individually for predicting operons. The gene pairs within operons have high fitness value by using these four scoring criteria, whereas those across transcription unit borders have low fitness value. On the other hand, it is easy to predict operons and makes the prediction ability stronger by using these four scoring criteria. The proposed method is examined on 683 known operons of *Escherichia coli K12* and an accuracy of 85.9987% is obtained.

1 Introduction

Operon is a string of one or more genes, which is transcribed as a fundamental unit and is on the same strand of a genomic sequence. Generally, genes within an operon have much shorter intergenic distances than genes at the borders of transcription units. And they usually belong to the same functional category, or show highly correlated expression patterns in microarray expression data [2], or their functions are related. Prediction of pathway often relies on operon, and the functions of operons reveal valuable information for studying protein function and drug design [1]. Therefore, understanding the operon maps of the whole genome is critical to reconstruction of regulatory networks and research on the whole genome.

Many computational algorithms for operon prediction using these properties have been developed in the past decade [2-12]. Salgado [4] developed an approach, which utilized the feature of intergenic distance between gene pairs within operon (WO pairs in short) and transcription unit borders (TUB pairs in short). Overbeek [5] proposed a method to search for conserved gene pairs across multiple genomes. Sabatti [7] used a

* Corresponding author.

Bayesian classification method to gene microarray expression data. With the increase of available data sources, Crave [8] developed naïve Bayes models, which used a rich variety of data types including sequence data, gene expression data, and functional annotations associated with genes to estimate the probability that a given sequence of genes constitutes an operon. Chen [9] developed an approach based on a comparative genomics approach and applied Neural Network to intergenic distance, COG function, phylogenetic profile. Jacob[11] proposed a Fuzzy guided genetic algorithm for operon prediction. This method used a genetic algorithm to evolve an initial population, which presents a putative operon map of a genome. Each putative operons are scored using intuitive fuzzy rules.

In this paper, we propose another method for assessing features by using different method, in which the four features, intergenic distance, participation in the same metabolic pathway, COG gene functions, microarray expression data are utilized. A Local Entropy Minimization method (LEM) is utilized for assessing the “fitness” of each adjacent gene pair based on intergenic distance. COG function log-likelihood is computed for adjacent gene pair. Correlation coefficient of microarray expression value is calculated. Genetic algorithm is used for evolving an initial population, each individual of which is created by clustering based on intergenic distances with different thresholds. We applied our method to *Escherichia coli K12* genome.

2 Data Representation

We have downloaded 319 completed Microbial Genomes data including 26 Archea and 293 Bacteria from GeneBank database (<http://www.ncbi.nlm.nih.gov/genomes/MICROBES/Complete.html>). From the RegulonDB database, we obtain 904 WO pairs and 653 TUB pairs for *E.coli*.

In this paper, our approach to predicting operon is based on genetic algorithm, in which we use intergenic distance, metabolic pathway, COG gene functions and microarray gene expression to compute the fitness value of each putative operon. A table $T: R \times A \rightarrow O$ is created from data files mentioned above, where A is the set of the four attributes, R is the number of adjacent gene pair, O is the set of 1, 2, 3, ..., k , k is the number of operons. If $O_i = O_{i+1}$, then $gene_i$ and $gene_{i+1}$ belong to an operon. Otherwise, they don't belong to an operon. Our object is to use the four properties of two adjacent gene pairs to predict if they belong to an operon. That is, we should compute the probability that two adjacent genes belong to a same operon. The pair-score of two adjacent genes should reflect the probability. The pair-scores for the four properties are computed as follows.

2.1 Intergenic Distance Features

We evaluate intergenic distance frequencies of known WO pairs and of known TUB pairs as Fig.1. For obtaining the pair-scores according to the intergenic distances, all the intergenic distance entropies are calculated according to Eq.(1).

$$E(p) = -p \log(p) - (1-p) \log(1-p) \quad (1)$$

where p is the probability that the intergenic distance d_i is a known WO. From Fig.1, we can see that the genes with smaller intergenic distances will be more possible to be within an operon. For showing the tendency, we propose a novel Local Entropy Minimization method to divide the set of intergenic distances into several intervals and compute the score of each interval by Eq.(2). Fig.2 presents the interval partition algorithm.

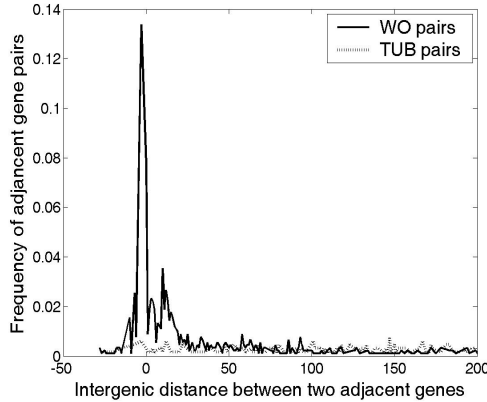


Fig. 1. Frequent intergenic distance distributions of adjacent gene pairs in operons in contrast with those of adjacent gene pairs at transcription unit boundaries

Algorithm: Interval Partition based on LEM

Input: Data array $D(P, 2)$ of P adjacent gene pairs.

Output: Intervals.

```

{
  Compute the entropy ( $d_i$ ) for every intergenic distance  $d_i, i \in [1, P]$ .
  Make an ordered list  $O$  of all of the adjacent gene pairs using intergenic distance  $d_i$ 
  from the largest to the smallest.
  Set an interval entropy threshold  $\Delta$ .
  Set the list  $\Theta$  to be an empty list.
  While(the list  $O$  is not empty)
  {
    Add the top intergenic distance in the list  $O$  to  $\Theta$ .
    Calculate the entropy from the first to the  $i$ th adjacent gene pair for  $i=2, 3, \dots, P$ .
    Choose the  $k$ th adjacent gene pair whose entropy is the smallest and add its
    intergenic distance to  $\Theta$ .
    Remove the record from the first to the  $k$ th from the list  $O$ .
  }
  Compute the score of each interval as defined in Eq.(2) and add them to  $\Theta$ .
  Return the list  $\Theta$ 
}

```

Fig. 2. Algorithm for the interval partition

At first, we choose an interval entropy threshold and sort the intergenic distances in descending order. Then from the beginning of the sorted intergenic distances, we take one distance out into an interval one by one and compute the interval entropy. If the entropy of the interval is larger than the threshold, then the process stops and we choose an interval from all of the generated intervals whose local entropy is minimum. We continue to find the next interval in the rest with the next point after the interval as the beginning. The rest intervals may be deduced by analogy. Thus, all the intergenic distances are partitioned into several intervals.

Since their entropy are equal if the probability of a known WO in an intergenic distance interval is equal to that of a known TUB in another interval, we define $S(d)$ to show their difference and consider it as the pair-score based on intergenic distance. $S(d)$ is defined in Eq.(2). $E(d)$ and $S(d)$ of intergenic distance between two adjacent genes are shown in Fig.3.

$$S(d) = \begin{cases} 1 - E(d) & d = [d_i, d_{i+1}], \text{ if } (P(WO | [d_i, d_{i+1}]) \geq \frac{1}{2}, \\ E(d) - 1 & d = [d_i, d_{i+1}], \text{ if } (P(WO | [d_i, d_{i+1}]) < \frac{1}{2}. \end{cases} \tag{2}$$

where $S(d)$ is the score of adjacent gene pair based on intergenic distance, $E(d)$ is the entropy of interval d .

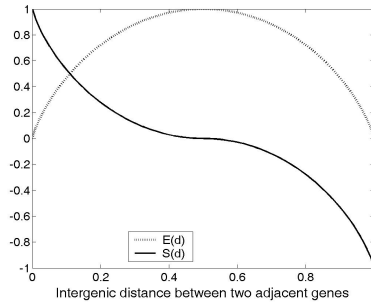


Fig. 3. $E(d)$ and $S(d)$ of intergenic distance between two adjacent genes

Table 1. 33 intervals of intergenic distance after using Local Entropy Minimization method

| Interval | Score | Interval | Score | Interval | Score | Interval | Score |
|-------------|----------|-----------|----------|-----------|----------|-----------|----------|
| [40611,561] | -1 | [380,560] | -0.3902 | [290,378] | -0.4007 | [235,289] | -0.2879 |
| [211,234] | -0.1502 | [186,210] | -0.0617 | [167,185] | -0.0148 | [140,166] | -0.1074 |
| [128,139] | -0.0817 | [114,127] | -0.0148 | [98,113] | -0.0192 | [91,97] | 0.1233 |
| [82,90] | 0.2081 | [77,81] | 0.0089 | [69,76] | 0.0573 | [65,68] | 0.0226 |
| [54,64] | 0.2317 | [49,53] | 0.029 | [47,48] | 0.188722 | [38,46] | 0.408327 |
| [32,37] | 0.374738 | [30,31] | 0.408327 | [28,29] | 0.456436 | [21,27] | 0.531 |
| [10,20] | 0.7355 | [5,9] | 0.5862 | [2,4] | 0.6549 | [-6,1] | 0.791443 |
| [-15,-7] | 0.66271 | [-18,-16] | 0.188722 | [-24,-19] | 0.278072 | [-70,-25] | 0.733235 |
| [-149,-71] | -1 | | | | | | |

From Fig.3, it can be seen that when the probability of a known WO in an intergenic distance interval is equal to 1, its entropy is equal to 0 and its pair-score equal to 1. When the probability of known TUB in an interval, its entropy is also equal to 0, but its pair-score is equal to -1. This shows that $S(d)$ can be used for prediction of operon.

We executed the algorithm and obtained 33 intervals. The intervals and their corresponding scores are presented in Table 1. From the table, we can see that the maximum score is appeared in the intergenic distance from -6 to 1.

2.2 Metabolic Pathway Features

Genes in one operon often carry out highly specific activity in a biochemical metabolic pathway [6, 12]. *E. coli*'s metabolic pathway data file is obtained from <http://ecocyc.org> and <http://www.genome.jp/kegg/>.

If $gene_i$ and $gene_j$ in the same operon are in the same metabolic pathway, then the pair-score $S_p(P_i)$ of the gene pair based on the metabolic pathway is 1 or else is 0.

2.3 COG Functions Features

We use the COGnitor program, which is available at <http://www.ncbi.nlm.nih.gov/COG/>, to create a protein function file by assigning a COG function category to each gene. Generally, genes in an operon have the same or similar COG functional category, which has been proved by Salgado [4] and Chen [9]. The COGs are clusters of orthologous groups, and consist of three main levels. In the first level, there are four classes: information storage and processing, cellular processing and signaling, metabolism and poorly characterized. There are many small classes in each first level class.

Table 2. Frequencies of adjacent pairs for different COG functional categories and their scores

| COG functional categories | Frequency WO pairs | Frequency TUB pairs | Log-likelihoods |
|------------------------------------|-----------------------|------------------------|-----------------|
| information storage and processing | 0.074 | 0.035 | 1.0733 |
| cellular processing and signaling | 0.132 | 0.040 | 1.7251 |
| metabolism | 0.463 | 0.182 | 1.3467 |
| Different COG categories | 0.319 | 0.722 | -1.1789 |

We also compute whether each adjacent gene pair has the same first-level functional category or not. For evaluating whether each adjacent gene pair belongs to a same operon or not, we compute the corresponding Log-Likelihood as defined in Eq.(3) for each the first level COG functional class, as shown in Table 2.

$$S_c(WO | class_i) = \log \frac{P(class_i | WO)}{P(class_i | TUB)} \quad (3)$$

where $class_i$ ($i=1, 2, 3, 4$) are the functional category as shown in Table 2.

2.4 Microarray Gene Expression Features

With the development of experiment technology, gene expression microarray experiments become more and more popular in eukaryote and prokaryote under different environment conditions. This makes microarray experiment data become available. DNA microarray data represents the expression intensities of genes. The genes in an operon are transcribed at the same level under many conditions. That is to say that the correlation coefficient of the expression value of genes in an operon should be equal to 1 in different microarray experiments. Hence, we can predict whether the genes are in an operon or not according to their correlation coefficient. Similar argument has been verified by Sabatti [7].

We have downloaded 8 microarray data files including GDS25, GDS95, GDS96, GDS97, GDS98, GDS99, GDS100, and GDS680 from The Gene Expression Omnibus (GEO) at NCBI. We calculate the Pearson correlation coefficient of gene x and gene y of n experiments as defined in Eq.(4).

$$r(x, y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (4)$$

where x_i is the log-ratio of gene x in the i th experiment, \bar{x} is the mean of x , y_i is the log-ratio of gene y in the i th experiment, \bar{y} is the mean of y , n is the number of experiments, in this paper n is 84. We take $r(x,y)$ as the score of microarray expression.

3 Operon Prediction Based on Genetic Algorithm

Genetic algorithm is used to find the optimization solution in this article and consists of three main processes as follows:

Firstly, create an initial population.

Secondly, compute the fitness of each chromosome.

Thirdly, use selection, crossover and mutation to evolve the population and create the next gap.

3.1 Initial Population

We create an initial population in which there are 20 chromosomes and each chromosome represents a possible solution that all genes are organized into operons according to their intergenic distance values under different thresholds. In this paper, the thresholds are between 0 and 300 bps. We make a Visual C++ program in which 20 intergenic distances with random thresholds are created and all genes are clustered into operons based on each different threshold. The intergenic distances and their operon numbers that is the order of operons are stored in 20 data files. For example, in *E.coli K12* genome, if a threshold is 150 bps, its gene cluster will be 1, 1, 1, 1, 2, 3, 4, 5, 6, 6, That is, the genes thrL, thrA, thrB and thrC belong to operon number 1,

and dnaK and dnaJ belong to operon number 2, etc. Each series of operon number in each data files is a chromosome in the initial population.

3.2 Fitness Function

The pair-score of adjacent genes is the sum of the pair-score based on the four features as aforementioned. To interpret how to calculate the fitness value of a putative operon, we suppose there are m genes and n gene pairs in the i th putative operon. The fitness value $fitness_i$ of the i th putative operon is as follows:

$$fitness_i = S(\bar{d}_i) + \frac{\sum_{i=1}^{m-1} \sum_{j=i+1}^m S_p(gene_i, gene_j)}{n} + \frac{\sum_{i=1}^{m-1} \sum_{j=i+1}^m S_c(gene_i, gene_j)}{n} + \frac{\sum_{i=1}^{m-1} \sum_{j=i+1}^m r(gene_i, gene_j)}{n}$$

where \bar{d}_i is the average of intergenic distances of the $m-1$ adjacent gene pairs in the putative operon, n is the number of all gene pairs in it. The fitness of a chromosome is the sum of the fitness of all the putative operons in the chromosome as following defined:

$$fitness = \sum_{i=1}^l fitness_i$$

where l is the number of operons in the chromosome.

3.3 Selection, Crossover and Mutation

After the initial population is generated, we compute the fitness value of each chromosome and sort these fitness values from the largest to the smallest. Then the classical roulette wheel method is utilized for selecting chromosomes for the next population. The roulette wheel method can make it more likely that there are many copies of the chromosomes with higher fitness values in the next population.

The crossover method used is the classical single-point crossover. For two chromosomes, a random crossover point is selected. Each of the two chromosomes is divided into two parts at the random position. Then we combine the first part of one chromosome with the second part of another chromosome into a new chromosome.

The mutation operator is carried out as following steps:

1. If two adjacent genes that are in the same strand and don't belong to one operon have very high score, the two adjacent genes will be combined.
2. If the last gene pair of a putative operon has very low score, we will remove the last gene from the operon.

For each generation, we compute its fitness based on the above method for each chromosome, and then carry out selection, crossover and mutation operation and

obtain the next generation, until the fitness values are stable in the successive generation. We select the best operons in the last generation as the final results.

4 Simulation Results

We develop a Visual C++ program to implement the proposed algorithm. To examine the performance of the method, we apply this method to the aforementioned *E. coli* K12 genome and obtain the best operons. Because the size of the predicted and experimental operons may be different, it is unreasonable to compare each operon. Therefore, we use the performance measurement that was used by many earlier researchers [9, 11, 13] to evaluate the proposed method. In the method, we only evaluate the known WO pairs and TUB pairs in the predicted operons. We take 904 known WO pairs as the positive test set and take 635 known TUB pairs as the negative test set.

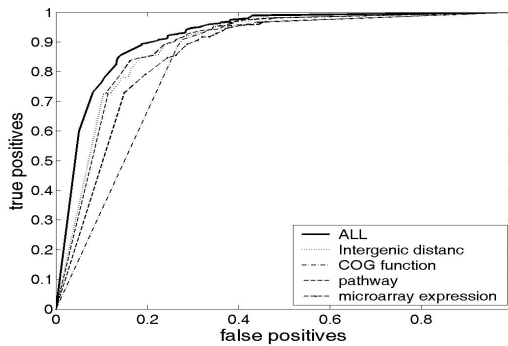


Fig. 4. ROC curves for *E. coli* based on each of the four features and all features

In order to obtain the best prediction result, we changed the very high score and the very low score in mutation operator. The accuracy of 85.9987% was obtained.

To assess the utility of different data sources in our method, we measured the predictive performance when each property is used alone and that involves all properties. Fig. 4 shows the ROC curve of the results of these experiments. We can see that the prediction based on all properties is much better than other four predictions based on the four properties individually, which proves the predicting ability of the presented method. A true positive rate of 90% with 20% false positives is obtained. In the four properties, intergenic distance and metabolic pathway are the most effective. COG function can also be used for predicting operons, but its false positive is very high. If metabolic pathway data were more enough, the prediction accuracy would be high.

5 Conclusions and Discussions

It has been reported to be an effective way for predicting operons by using different kinds of biological information. In contrast to the conventional methods where all

kinds of biological information are dealt with using the same method, we apply different approaches to different genome information for exploiting their unique biological characteristics. A modified genetic algorithm is presented by defining the fitness function with four kinds of genome information involved. Numerical experimental results show that the prediction accuracy of *Escherichia coli* K12 is improved efficiently. Future research of this topic is to apply this method to other genomes and add other genome information to further improve the accuracy.

Acknowledgement

The authors would like to thank members of Bioinformatics Group of JLU and UGA for the invaluable assistance and discussions. The authors are grateful to the support of the National Natural Science Foundation of China (60433020, 60673023, 60673099), the science-technology development project of Jilin Province of China (20050705-2), the European Commission under grant No. TH/Asia Link/010 (111084), "985" project of Jilin University of China and Science Foundation for Yong Teachers of Northeast Normal University (20050105).

References

1. Yeh, P., Tschumi, A.I., Kishony, R.: Functional classification of drugs by properties of their pairwise interactions. *Nature Genetics*. 38 (2006)489-494
2. Chen, X., Su, Z., Dam, P., Palenik, B., Xu, Y., and Jiang, T.: Operon prediction by comparative genomics: An application to the *Synechococcus* sp. WH8102 genome. *Nucleic Acids Res.* 32 (2004) 2147–2157
3. Yada, T., Nakao, M., Totoki, Y., and Nakai, K.: Modeling and predicting transcriptional units of *Escherichia coli* genes using hidden Markov models. *Bioinformatics*.15 (1999) 987–993
4. Salgado, H., Moreno-Hagelsieb, G., Smith, T. and Collado-Vides, J.: Operons in *Escherichia coli*: genomic analyses and predictions. *Proc Natl Acad. Sci.* 97 (2000) 6652-6657
5. Overbeek, R., Fonstein, M., D'Souza, M., Pusch, G.D. and Maltsev, N.: The use of gene clusters to infer functional coupling. *Proc. Natl. Acad. Sci.* 96(1999) 2896-2901
6. Zheng, Y., Szustakowski, J.D., Fortnow, L., Roberts, R.J., and Kasif, S.: Computational identification of operons in microbial genomes. *Genome Res.* 12(2002)1221–1230
7. Sabatti, C., Rohlin, L., Oh, M.K., and Liao, J.C.: Co-expression pattern from DNA microarray experiments as a tool for operon prediction. *Nucleic Acids Res.* 30 (2002) 2886–2893
8. Craven, M., Page, D., Shavlik, J., Bockhorst, J., and Glasner, J.: A probabilistic learning approach to whole-genome operon prediction. *Proc. 8th International Conference on Intelligent Systems for Mol. Biol.* (2000) 116–127
9. Chen, X., Su, Z.C., Xu, Y., Jiang, T.: Computational Prediction of Operons in *Synechococcus* sp. WH8102. *Genome Informatics.* 15 (2004) 211-222
10. Dam, P., Olman, V., Xu, Y.: Improving Operon Prediction in *E. coli*. 2005 IEEE Computational Systems Bioinformatics Conference - Workshops (CSBW). (2005) 69-70
11. Jacob, E., Sasikumar, R., Nair, K. N. R.: A fuzzy guided genetic algorithm for operon prediction *Bioinformatics.* 21 (2005) 1403-1407

12. Ogata, H., Fujibuchi, W., Goto, S., and Kanehisa, M.: A heuristic graph comparison algorithm and its application to detect functionally related enzyme clusters. *Nucleic Acids Res.* 28 (2000) 4021–4028
13. Moreno-Hagelsieb, G., Collado-Vides, J.: A powerful non-homology method for the prediction of operons in prokaryotes. *Bioinformatics.* 18 (2002) 329–336

Application of Micro-GA for an Optimal Direct Design Method of Steel Frame

Se-Hyu Choi

Department of Civil Engineering, Kyungpook National University
Daegu, 702-701, South Korea
shchoi@knu.ac.kr

Abstract. In this paper, an optimal direct design method of steel frame using advanced analysis and genetic algorithm is presented. The advanced analysis realistically assesses both strength and behavior of a structural system and its component members in a direct manner. The micro-GA is used for minimum weight optimization of steel frames. Constraint functions are load-carrying capacities and serviceability. The optimum designs determined by the proposed method are lighter than those given by other approaches.

1 Introduction

The steel design methods used in the U.S. are Allowable Stress Design (ASD), Plastic Design (PD), and Load and Resistance Factor Design (LRFD). However, despite popular use of conventional design methods as a basis for design, these design methods have their major limitations. The first of these is that it does not give an accurate indication of the factor against failure, because it does not consider the interaction of strength and stability between the member and structural system in a direct manner. The second limitation is probably the rationale of the current two-stage process in design: elastic analysis is used to determine the forces acting on each member of a structure system, whereas inelastic analysis is used to determine the strength of each member treated as an isolated member. There is no verification of the compatibility between the isolated member and the member as part of a structural system.

With the development of computer technology, two aspects, the stability of separate members, and the stability of the structure as a whole, can be treated rigorously for the determination of the maximum strength of the structures. The development of the direct approach to design is called "Advanced Analysis". In this direct approach, there is no need to compute the effective length factor, since separate member capacity checks encompassed by the specification equations are not required.

The purpose of this paper is to present an optimal and direct design method of steel frames using advanced analysis and genetic algorithms. Until now, several advanced analyses for steel frames were developed by Ziemian et al. [1], Prakash and Powell [2], Liew and Tang [3], and Kim and Choi [4]. When these advanced analyses are combined with optimal design procedures leading to minimum weight, the results must be a considerable contribution to practicing engineering.

2 Advanced Analysis for Direct Design Method

From Kim and Choi [4], the force-displacement relationship of the beam-column member considering the second-order effects associated with P- δ and P- Δ moments and gradual yielding due to residual stresses and flexure may be expressed as

$$\begin{Bmatrix} P \\ M_{yA} \\ M_{yB} \\ M_{zA} \\ M_{zB} \\ T \end{Bmatrix} = \begin{bmatrix} \frac{E_t A}{L} & 0 & 0 & 0 & 0 & 0 \\ 0 & k_{iyy} & k_{jyy} & 0 & 0 & 0 \\ 0 & k_{ijy} & k_{jly} & 0 & 0 & 0 \\ 0 & 0 & 0 & k_{iiz} & k_{jiz} & 0 \\ 0 & 0 & 0 & k_{ijz} & k_{jlz} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{GJ}{L} \end{bmatrix} \begin{Bmatrix} \delta \\ \theta_{yA} \\ \theta_{yB} \\ \theta_{zA} \\ \theta_{zB} \\ \phi \end{Bmatrix} \quad (1)$$

where

$$k_{iyy} = \eta_A \left(S_1 - \frac{S_2^2}{S_1} (1 - \eta_B) \right) \frac{E_t I_y}{L}, \quad k_{jyy} = \eta_A \eta_B S_2 \frac{E_t I_y}{L}, \quad k_{ijy} = \eta_B \left(S_1 - \frac{S_2^2}{S_1} (1 - \eta_A) \right) \frac{E_t I_y}{L} \quad (2a,c)$$

$$k_{iiz} = \eta_A \left(S_3 - \frac{S_4^2}{S_3} (1 - \eta_B) \right) \frac{E_t I_z}{L}, \quad k_{jiz} = \eta_A \eta_B S_4 \frac{E_t I_z}{L}, \quad k_{ijz} = \eta_B \left(S_3 - \frac{S_4^2}{S_3} (1 - \eta_A) \right) \frac{E_t I_z}{L} \quad (2d,f)$$

The terms η_A and η_B are scalar parameters that allow for gradual inelastic stiffness reduction of the element associated with plastification at end A and B. This term is equal to 1.0 when the element is elastic, and zero when a plastic hinge is formed. S_1 , S_2 , S_3 , and S_4 are the stability functions with respect to y and z axes, respectively. E_t is the CRC tangent modulus to account for gradual yielding along the length of axially loaded members between plastic hinges.

3 Optimum Design Problem

One of the most important considerations in the optimization of steel frames is that frame members are generally to be selected from available steel profiles. The simple GA introduced by Holland [5] has a better chance in finding optimal solutions with the discrete design variables and is still widely used in the multidisciplinary field of engineering design optimization. If the population size is kept smaller to reduce operation time, its search performance becomes unsatisfactory due to the genetic drift of a specified bit. In this paper, micro-GA developed by Krishnakumar [6] is adopted for the efficient use of population while keeping it small [7].

3.1 Fitness Function

The GAs are suitably applied in the unconstrained maximization problems. However, the design optimization problems are constrained minimization problems. The design

optimization problems should be transformed into unconstrained maximization problems to use the genetic algorithms. In this paper, the fitness function is defined in Eq.(3), where the weight function for a steel frame has been modified using the violations of normalized constraints.

$$F(x) = K - OBJ(1 + rC) \tag{3}$$

in which, K is any number that is large enough to transform the constrained minimization problem to an unconstrained maximization problem, OBJ is total weight of structure, r is an increment parameter of 30 for penalty function, and C is the coefficient of violation.

$$K = \sum_{i=1}^n 1000L_i \tag{4}$$

$$OBJ = \sum_{i=1}^n W_i L_i \tag{5}$$

$$C = \sum_{i=1}^m \max(0, g_i^+(x)) \tag{6}$$

where W_i is the unit weight of AISC WF-shape used and $g_i^+(x)$ is a i -th inequality constraint condition which has positive values.

3.2 Constraint Condition by Load Resistant Capacity and Serviceability

With using load-ratio acting on structure, constraint condition by load resistance capacity is formulated as

$$g_1 = \frac{1.0}{\phi} - \lambda \leq 0 \tag{7}$$

where ϕ is resistance factor of 0.9 for the structure collapsed by yielding and 0.85 by buckling. λ is the applied load ratio of structure. According to studies of Ad Hoc committee [8] and Ellingwood [9], this constraint condition by serviceability is formulated as

$$g_2 = \frac{(\Delta_{bv})_i}{L_i / 360} - 1.0 \leq 0 \tag{8}$$

$$g_3 = \frac{(\Delta_{cv})_j}{H_j / 300} - 1.0 \leq 0 \tag{9}$$

Here, Eq.(8) and Eq.(9) are constraint conditions for deflection and story drift, respectively. L_i and $(\Delta_{bv})_i$ are length and deflection of i -th beam. H_j and $(\Delta_{cv})_j$ mean height of column and story drift.

3.3 Optimum Design Aalgorithm

The GA-based optimum design algorithm developed for direct design method of steel frames consists of the following step.

1. Initial population is constructed randomly. The size of initial population is selected as 10. The maximum number of generations is initially taken as 1000.
2. For each individual, the binary codes for all design variables are converted into a base-10 sequence number that identifies the corresponding sections.
3. The frame is analyzed for these sections with the proposed advanced analysis and the applied load ratio is obtained under the factored load and the deflection and the story drift are obtained under service load.
4. Using Eq.(3), the fitness value for each individual is calculated.
5. Two individuals are selected arbitrarily and two off-springs are generated by the tournament selection to manage small population efficiently.
6. Two individuals are coupled randomly and two off-springs are generated using a single point crossover operation, the value of 1.0 is used for the probability of crossover. The new population is obtained.
7. When applying the micro-genetic algorithm, the mutation is not required.
8. The individual that satisfies the design constraints and has the minimum weight is recorded.
9. The initial population is replaced by the new population and steps 1-8 are repeated.

4 Design Examples

4.1 Planar Three-Story Frame

Figure 1 shows the topology and loading of a two-bay three-story frame. This frame was designed by Pezeshk et al. [10] and Yun and Kim [11]. Structural elements are classified into two groups as shown in Figure 2. Nine column members and six beam members are required to have the same WF-shape in the design, respectively. The yield stress used is 250 MPa (36 ksi) and Young's modulus is 200,000 MPa

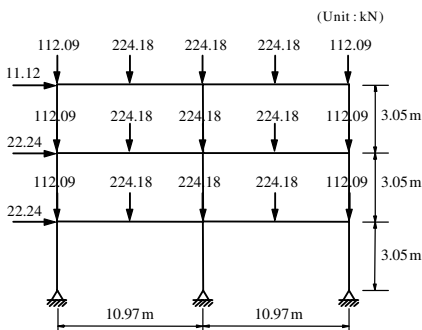


Fig. 1. Planar three-story frame

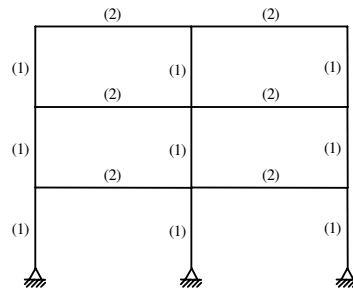


Fig. 2. Design variables

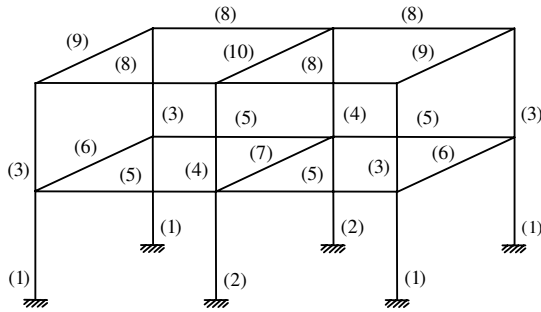


Fig. 4. Design variables

principal axes are parallel with the x -axis. The structural members are classified into eight groups as shown in Figure 4. The yield stress and Young's modulus are taken to be 250 MPa (36 ksi) and 200,000 MPa (29,000 ksi), respectively. The further reduced tangent modulus modeling proposed by Kim and Chen [12] is used to account for geometric imperfections.

The optimized design for the frame obtained by the proposed method is compared with that generated by Kim and Ma [13] as shown in Table 2. The frame encountered ultimate state when the applied load ratio reached 1.16. Since the frame collapses by forming plastic mechanism, the system resistance factor of 0.9 is used. The ultimate load ratios λ resulted in 1.044 (0.9×1.16), which was greater than 1.0 and the member sizes of the system are adequate. The weight of optimized design for the frame is 25.6% lighter than that of Kim's design. The number of nonlinear analysis

Table 2. Comparison of member sizes of space three-story frame

| Design Variables | Proposed | Kim and Ma |
|--------------------|--------------------------|--------------------------|
| Group 1 | W10×54 | W14×145 |
| Group 2 | W12×152 | W16×67 |
| Group 3 | W6×16 | W10×112 |
| Group 4 | W27×217 | W12×65 |
| Group 5 | W12×45 | W24×68 |
| Group 6 | W24×62 | W16×31 |
| Group 7 | W12×22 | W12×30 |
| Group 8 | W14×34 | W10×68 |
| Group 9 | W10×30 | W8×18 |
| Group 10 | W12×22 | W12×30 |
| Total weight | 111.25 kN (25,010 lb) | 145.58 kN (32,727 lb) |
| Applied load ratio | 1.16 | 1.93 |
| Number of analysis | 930 | 200,000 |

required by the proposed method was 930 times in optimum design for the frame. However, the number of nonlinear analysis required to get optimum value by Kim and Ma was 200,000 since a population size of 10 and a generation size of 20,000 were used to find optimum value. Therefore, the proposed optimum design proves its computational efficiency.

5 Conclusions

In this paper, an optimal design of steel frames using genetic algorithms and advanced analysis is developed. The summaries and conclusions of this study are as follows.

The weight of optimized design for the planar three-story frame is 32.8% and 13.9% lighter than those of Pezeshk's design and Yun's design, respectively. The weight of optimized design for the space two-story frame is 25.6% lighter than that of Kim's design.

The proposed optimum design method for the planar three-story frame can save 56 and 8 analysis times compared with the method proposed by Pezeshk et al. and Yun and Kim, respectively. The proposed optimum design method for the space two-story frame can save 215 analysis times compared with the method proposed by Kim and Ma.

The advanced analysis and optimal design method are combined to provide much benefit to practical engineering.

Acknowledgments. This work was supported by the Brain Korea 21 Project in 2006. The financial support is gratefully acknowledged.

References

1. Ziemian, R.D., McGuire, W., and Dierlein, G.G.: Inelastic limit states design part II: three-dimensional frame study, *ASCE J. Struct. Eng.*, 118(9) (1992) 2550-2568.
2. Prakash, V. and Powell, G.H.: DRAIN-3DX: Base program user guide, version 1.10, A Computer Program Distributed by NISEE / Computer Applications, Department of Civil Engineering, University of California, Berkeley (1993)
3. Liew, J.Y. and Tang, L.K.: Nonlinear refined plastic hinge analysis of space frame structures, Research Report No. CE027/98, Department of Civil Engineering, National University of Singapore, Singapore. (1998)
4. Kim, S.E. and Choi, S.H.: Practical advanced analysis for semi-rigid frames, *Int. J. of Soldis and Structures*, 38, (2001) 9111-9131.
5. Holland, K.: *Adaptation in natural and artificial systems*, The University of Michigan Press, Ann Arbor, MI.
6. Krishnakumar, K.: *Micro-genetic algorithms for stationary and non-stationary function optimization*, SPIE, Intelligent Control and Adaptive Systems, Vol. 1196. (1989)
7. Ryu, Y.S., Kim, J.H., Cho, H.M., and Kim, J.K.: LRFD based design optimization of steel box girder sections using genetic algorithm, *KSCE*, Vol. 6, No 2 (2002) 127-134.
8. Ad Hoc Committee on Serviceability Structural serviceability: a critical appraisal and research needs, *J. Struct. Eng.*, ASCE, 112(12) (1986) 2646-2664.

9. Ellingwood, B.: Serviceability guidelines for steel structures, *Engineering Journal*, AISC, 26, 1st Quarter (1989) 1-8.
10. Pezeshk, S., Camp, C.V. and Chen, D.: Design of nonlinear framed structures using genetic optimization, *J. Struct. Eng.*, ASCE, 126(3), (2000) 382-388.
11. Yun, Y.M. and Kim, B.H.: Optimum design of plan steel frames using second-order inelastic analysis and a genetic algorithm, *KSCE Journal of Civil Engineering*, KSCE, 24(1-A) (2004) pp.87-100.
12. Kim, S.E. and Chen, W.F.: Practical advanced analysis for unbraced steel frame design, *J. Struct. Eng.*, ASCE, 122(11). (1996) 1259-1265.
13. Kim, S.E. and Ma, S.S.: Nonlinear inelastic optimal design using genetic algorithm, *KSCE Journal of Civil Engineering*, KSCE, 23(5-A) (2003) 841-850.

Multi-objective Optimal Public Investment: An Extended Model and Genetic Algorithm-Based Case Study*

Lei Tian¹, Liyan Han¹, and Hai Huang²

¹ School of Economics and Management
Beihang University, Beijing 100083, P.R. China
tianlei@sem.buaa.edu.cn

² School of Computer Science and Engineering
Beihang University, Beijing 100083, P.R. China
huanghai@vrlab.buaa.edu.cn

Abstract. Under the multi-region and multi-sector consideration, the previous double-objective optimal public investment model is extended to involve optimal employment rate objective and time-flow total income maximization objective first. Then genetic algorithm is applied to solve the multi-objective model. Finally a case study is carried out to verify the superiority of the genetic algorithm-based solution to traditional public investment distribution approach.

1 Introduction

The governments usually adopt multiple financial tools to achieve multiple goals when implementing optimal policies related to public investment. The available variables include tax rates, transfer payments, regional proportions of public investment, etc.; the goals may be multiple, including economic growth [1], equity and efficiency [2], full employment [3], etc. The former optimal public investment model [4] can be extended to involve two new objectives - the optimal employment rate objective and time-flow total income maximization objective. With these objectives added, the model may be more practical. And the more complicated model can be solved with genetic algorithm.

2 Literature Review

First, Single-objective two-region two-sector optimal economic growth model and double-objective multi-region multi-sector optimal public investment model are briefed in this section.

* This paper is supported by the Innovation Foundation of BUAA for PhD Graduates and the Specialized Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20050006025).

2.1 Single-Objective Two-Region Two-Sector Optimal Economic Growth Model

The optimal public investment model is derived from optimal economic growth model. Suppose the aggregative production function is homogenous production function of degree one: $Y(t) = F(K(t), L(t))$, and the output-capital ratios of private and public sector are constant. The basic objective is to maximize final total income $Y(T)$. As for the constraints, the economy is divided into two sectors (public and private sector) and two regions (1 and 2). The production function is assumed as linear. Under different rates of savings (s_i) and different distribution rates of investment (φ and ϕ) from public and private sectors, considering income tax rate $r(t)$, the model can be summarized as follows (Sakashita Model [5]):

$$\begin{aligned}
 &Max\ W = Y_1(T) + Y_2(T) \\
 &s.t.\ Y_1'(t) = \sigma_1\phi(1 - r(t))(s_1Y_1 + s_2Y_2) + \delta_1\varphi r(t)(s_1Y_1 + s_2Y_2) \\
 &\quad Y_2'(t) = \sigma_2(1 - \phi)(1 - r(t))(s_1Y_1 + s_2Y_2) + \delta_2(1 - \varphi)r(t)(s_1Y_1 + s_2Y_2) \\
 &\quad 0 \leq r \leq \theta, \varphi \in [0, 1], Y_1(0) = Y_1^0, Y_2(0) = Y_2^0
 \end{aligned}$$

σ_i and δ_i are output-capital ratios of private and public sectors in region i .

2.2 Double-Objective Multi-region Multi-sector Optimal Public Investment Model

After TIAN, etc. [4] introducing multi-region multi-sector Cobb-Douglas production function and labor-investment ratio, the former assumptions are relaxed. Considering cross-region income per capita gap minimization, modifying taxation factor and capital transfer loss, the optimal economic growth model maybe extended to a double-objective multi-constraints optimal problem from the public investment perspective:

$$Max\ W = \sum_{i=1}^n \sum_{j=1}^m \omega_i \xi_{ij} Y_{ij}(T)$$

$$E = (-1) \sum_{k,v=1}^n \int_{T_0}^T |Y_k(t)/N_k(t) - Y_v(t)/N_v(t)| dt \tag{1}$$

$$s.t.\ I(t) = K'(t) + \gamma K(t) \tag{2}$$

$$L_{ij}(t) = \lambda_{ij} I_{ij} + C_{ij}, \lambda_{ij} > 0, C_{ij} > 0 \tag{3}$$

$$\begin{aligned}
 K'(t) = &r(t) \sum_{i=1}^n [(1 - \sum_{k=1}^n b_{ik}) z_i \sum_{j=1}^m \varphi_{ij} Y_{ij}(t)] \\
 &+ (1 - r(t)) \sum_{i=1}^n [(1 - \sum_{k=1}^n a_{ik}) s_i \sum_{j=1}^m \phi_{ij} Y_{ij}(t)] \\
 &- \gamma \sum_{i=1}^n \sum_{j=1}^m K_{ij}(t)
 \end{aligned} \tag{4}$$

$$Y(t) = \sum_{i=1}^n \sum_{j=1}^m A_{ij} K_{ij}(t)^{\alpha_{ij}} L_{ij}(t)^{\beta_{ij}}. \tag{5}$$

φ_{ij} , the weight of public sector investment to sector j of region i , should be solved. The relevant parameters are: $I(t)$ is the investment; $K(t)$ is the capital stock, and $K'(t) = \frac{dK(t)}{dt}$; the current capital stock depreciates at a constant rate γ ; A_{ij} is the contribution of technological innovation to output of sector j of region i ; α_{ij}/β_{ij} means 1% increase of capital/simple labor will bring forth α_{ij}/β_{ij} increase of output; s_i/z_i is rate of savings of private/public sector of region i ; ξ_{ij} is weight of sector j of region i ; ω_i is weight of region i ; ϕ_{ij} is weight

of private sector investment to sector j of region i ; a_{ik}/b_{ik} is the proportion of capital transfer loss between regions; λ_{ij} is the labor-investment ratio of sector j of region i ; C_{ij} is the necessary simple labor of sector j in region i ; N_i is the population in region i ; $Y_{ij}(0)$ is the initial income; $I(0)$ is the initial total investment. Please refer to the previous study of TIAN, etc. [4] for GA-based solution and case study verifying the validity of the model and solution.

3 New Objective Functions

In this section, time-flow total income maximization and employment rate maximization objectives will be discussed. The control variables include rate of savings, tax rates, public investment shares of regions and sectors, etc. And the new model - multi-objective public investment model can be established.

3.1 Time-Flow Total Income

The maximization objective of final total income does not explain the total income level during the whole planning horizon. In order to overcome this defect of the model, Intriligator (1964), Takayama (1967), Friesz (1977), etc. pioneered in introducing time flow objective to the economic growth model. The time flow objective will be redesigned to fit into the optimal public investment model by taking relative importance of the regions and sectors into consideration.

Suppose μ is the exponential discounting factor. In the period (T_0, T) , if the total welfare is figured with the discounted total income μ , the maximization objective of discounted time-flow income can be written as: $Max W = \int_{T_0}^T e^{-\mu(t-T_0)} Y(t) dt = \int_{T_0}^T e^{-\mu(t-T_0)} \sum_{i=1}^n \sum_{j=1}^m \omega_i \xi_{ij} Y_{ij}(t) dt$.

If the final total income maximization objective and time-flow total income maximization objective are combined, they can be used to balance the relative importance of final total income and discounted time-flow income, and the weighted average of the two values can figure the maximization objective of total welfare:

$$Max W = \eta \sum_{i=1}^n \sum_{j=1}^m \omega_i \xi_{ij} Y_{ij}(T) + (1 - \eta) \int_{T_0}^T e^{-\mu(t-T_0)} \left(\sum_{i=1}^n \sum_{j=1}^m \omega_i \xi_{ij} Y_{ij}(t) \right) dt$$

3.2 Moderate Employment Rate

Full employment is one of the four basic macroeconomic objectives. In order to maximize total employment rate, it is only necessary to:

$$Max P = \frac{\sum_{i=1}^n L_i(t)}{\sum_{i=1}^n N_i(t)} \tag{7}$$

Considering equity between regions, a moderate employment rate should be pursued. Then a lower limit B of regional employment rate can be set, so that:

$$\frac{L_i(t)}{N_i(t)} \geq B, 0 < B < 1 \tag{8}$$

There is certain relationship between these objectives. For example, income maximization and full employment are constrained mutually. In addition, these three objectives must be realized by meeting certain constraints.

3.3 Multi-objective Model

The multi-objective optimal public investment model is:

$$\text{Max (1), (6), (7), s.t. (2), (3), (4), (5), (8) .} \tag{9}$$

The dependent variable of the model is the investment proportion of sector j of region i . The control variables of the model are financial tools, including tax rate, weights of every sector of every region, etc. The main external parameters are rate of savings, private sector investment proportions, discounting rate, etc.

4 GA-Based Solution

Genetic algorithm is the effective algorithm designed specially for complicated optimization problems on the basis of biological simulation, the powerful stochastic searching optimization technique [7], [8], and has been applied in solving optimal public investment problem [9]. As for the nonlinear [10] optimal public investment model with many variables, multiple objective functions, and multiple constraints in this paper, objective functions and constraints are handled to adapt the model into the multi-objective multi-constraint programming model suitable for genetic algorithm-based solution. Then encoding and decoding approaches are designed according to characteristics of the model.

4.1 Fitness Function

Resolve ordinary differential equation (2):

$$K(t) = e^{-\gamma t} \left(\int I(t)e^{\gamma t} dt + \tilde{C} \right) . \tag{10}$$

Combining (3), (5) and (10), we get:

$$Y_{ij}(t) = A_{ij} K_{ij}^{\alpha_{ij}}(t) L_{ij}^{\beta_{ij}}(t) = A_{ij} (e^{-\gamma t} (\int I_{ij}(t)e^{-\gamma t} dt + \tilde{C}))^{\alpha_{ij}} (\lambda_{ij} I_{ij} + C_{ij})^{\beta_{ij}} .$$

Without losing generality, suppose the government is making public investment plan for period (T_0, T) at T_0 . So at T_0 the government plans to invest $\varphi_{ij} I$ in sector j of region i during (T_0, T) .

Definition: The governmental investment $I_{ij}(t)$ during (T_0, T) is called the feasible investment path of sector j of region i .

Proposition: There are infinite feasible investment paths $(I_{ij}(t))$ for given T_0, T, φ_{ij} , and I .

Proof: According to the definition of feasible investment path, the following formula exists:

$$\varphi_{ij}I = \int_{T_0}^T I_{ij}(t)dt . \tag{11}$$

among which the integral indicates the cumulative sum of $I_{ij}(t)$.

First, there obviously exists a feasible investment path $I_{ij}(t)$ satisfying formula (11), such as

$$I_{ij}(t) = \frac{\varphi_{ij}I}{T-T_0}$$

Second, suppose $p(t)$ is a viable feasible investment path, i.e. $\varphi_{ij}I = \int_{T_0}^T p(t)dt$. Then for any real number v_1 and v_2 , $v_1 \leq p(t)$ and $v_1 = \frac{(v_2-1)\varphi_{ij}I}{(T-T_0)}$: $\int_{T_0}^T \frac{p(t)+v_1}{v_2} dt = \varphi_{ij}I$.

So $\frac{p(t)+v_1}{v_2}$ is also a feasible investment path. Since v_1 and v_2 can be any real number, there are infinite feasible investment paths. The proof of Proposition 1 is completed.

Although there are infinite feasible investment paths of sector j of region i , most investment processes can be divided into two stages in practice: At first a initial \widetilde{I}_{ij} is invested, and the rest is invested gradually. The investment process can be described with the following segmented function:

$$I_{ij}(t) = \begin{cases} \widetilde{I}_{ij} & \text{if } t = T_0 \\ (\varphi_{ij}I - \widetilde{I}_{ij})/(T - T_0) & \text{if } t > T_0 \end{cases} . \tag{12}$$

Then, formula (1), (6), and (7) can be rewritten as:

$$E = (-1) \sum_{k,v=1}^n \int_{T_0}^T \left| \frac{\sum_{j=1}^m A_{kj}(e^{-\gamma t}(\int I_{kj}(t)e^{\gamma t} dt + \widetilde{C}))^\alpha (\lambda_{kj}I_{kj} + C_{kj})^\beta}{N_k(t)} - \frac{\sum_{j=1}^m A_{vj}(e^{-\gamma t}(\int I_{vj}(t)e^{\gamma t} dt + \widetilde{C}))^\alpha (\lambda_{vj}I_{vj} + C_{vj})^\beta}{N_v(t)} \right| dt \tag{13}$$

$$W = \eta \sum_{i=1}^n \sum_{j=1}^m \omega_i \xi_{ij} A_{ij} (e^{-\gamma t}(\int I_{ij}(t)e^{\gamma t} dt + \widetilde{C}))^\alpha (\lambda_{ij}I_{ij} + C_{ij})^\beta + (1 - \eta) \int_{T_0}^T e^{-\mu(t-T_0)} (\sum_{i=1}^m \sum_{j=1}^m \omega_i \xi_{ij} A_{ij} (e^{-\gamma t}(\int I_{ij}(t)e^{\gamma t} dt + \widetilde{C}))^\alpha (\lambda_{ij}I_{ij} + C_{ij})^\beta) dt \tag{14}$$

$$P = \frac{\int_{T_0}^T \sum_{i=1}^n \sum_{j=1}^m (\lambda_{ij}I_{ij}(t) + C_{ij}) dt}{\int_{T_0}^T \sum_{k=1}^n N_k(t) dt} \tag{15}$$

Combining (12), (13), (14), and (15), it can be seen that optimal public investment problem is actually to solve $Max(E, W, P)$, i.e. to solve φ_{ij} - the proportion of investment in sector j of region i to the total investment. After the constraints are handled as above, some constraints have been incorporated into the objective functions, and only (4) and (8) are left as constraints of optimal public investment problem. Since E, W, P , and constraints (4) and (8) are the all and only functions of φ_{ij} , they can be rewritten as the following equivalent formula:

$$\begin{aligned}
 E &= f_1(\varphi_{11}, \varphi_{12}, \dots, \varphi_{1m}, \varphi_{21}, \dots, \varphi_{2m}, \dots, \varphi_{n1}, \dots, \varphi_{nm}) \\
 W &= f_2(\varphi_{11}, \varphi_{12}, \dots, \varphi_{1m}, \varphi_{21}, \dots, \varphi_{2m}, \dots, \varphi_{n1}, \dots, \varphi_{nm}) \\
 P &= f_3(\varphi_{11}, \varphi_{12}, \dots, \varphi_{1m}, \varphi_{21}, \dots, \varphi_{2m}, \dots, \varphi_{n1}, \dots, \varphi_{nm}) \quad , \\
 f_4(\varphi_{11}, \varphi_{12}, \dots, \varphi_{1m}, \varphi_{21}, \dots, \varphi_{2m}, \dots, \varphi_{n1}, \dots, \varphi_{nm}) &= 0 \\
 f_5(\varphi_{11}, \varphi_{12}, \dots, \varphi_{1m}, \varphi_{21}, \dots, \varphi_{2m}, \dots, \varphi_{n1}, \dots, \varphi_{nm}) &\leq 0
 \end{aligned}$$

among which $f_6 = \sum_{i=1}^n \sum_{j=1}^m \varphi_{ij} = 1$.

When this multi-objective programming problem is solved with genetic algorithm, constraints f_4 and f_6 should be relaxed properly to avoid premature convergence of the algorithm, so:

Let $f_7 = |f_4| \leq \Delta_1$, $f_8 = |f_6 - 1| \leq \Delta_2$, among which Δ_1 and Δ_2 are positive real numbers with very small value. Then the optimal public investment model can be rewritten as:

$$\text{Max}(E, W, P) \quad \text{s.t.} \quad f_5 \leq 0, f_7 \leq \Delta_1, f_8 \leq \Delta_2 \quad . \tag{16}$$

Combine weighted-sum approach and penalty function approach, transform (16) into the following single objective programming model:

$$\text{Max } V = \pi_1 E + \pi_2 W + \pi_3 P + F(f_5) + F(f_7) + F(f_8) \quad , \tag{17}$$

among which $\pi_i \geq 0 (i = 1, 2, 3)$ is the weight of three optimal objective functions in (17), and $\sum_{i=1}^3 \pi_i = 1$. $F(f_5)$, $F(f_7)$ and $F(f_8)$ are negative penalty functions [11] which penalize the solutions violate constraints.

4.2 Encoding

It is a key to GA to encode and decode the solutions into chromosomes. In the optimal cross-region public investment model, a binary string [12] of 7 bits can be used to represent φ_{ij} . The value of the binary string is a percent. If it exceeds 100%, the value is defined as 100%. For example, if the corresponding binary string of φ_{11} in the chromosome is 0011001, then $\varphi_{11} = 15\%$. Then after encoding $m \times n$ variables, the chromosomes of optimal public investment model are binary strings with the length of $7 \times m \times n$.

5 Case Study

The validity of multi-region multi-sector public investment model and GA-based solution has been proven in previous study [4]. In this section, a new case study is carried out to verify the superiority of GA-based solution to traditional investment distribution approaches based on negotiation.

Suppose there are two regions with two sectors each. Sector 2 is more labor-intensive than sector 1. Region 2 is less developed than region 1. The relevant parameters are:

$A = 5$, representing the contribution of technological innovation to output;

$\alpha = 0.6$, representing 1% increase of capital will bring forth 60% increase of output;

$\beta = 0.4$, representing 1% increase of simple labor will bring forth 40% increase of output;

$s_i = 50\%$, $z_i = 90\%$, representing rates of savings of private sector and public sector;

$\mu = 0.9$, representing the discount rate of income;

$\xi_{11} = 0.6, \xi_{12} = 0.4, \xi_{21} = 0.7, \xi_{22} = 0.3$, representing weights of the two sectors in the two regions;

$\omega_1 = 0.4, \omega_2 = 0.6$, representing weights of the two regions;

$\eta = 0.2$, representing the weight of discounted time-horizontal total income in total income;

$\phi_{11} = \phi_{12} = \phi_{21} = \phi_{22} = 0.25$, representing the weights of private sector investment to sector j of region i ;

$r(t) = 0.3$, representing the average tax rate;

$a_{12} = a_{21} = b_{12} = b_{21} = 0.2$, representing the proportion of capital transfer loss between regions;

$\gamma = 0.1$, representing the capital stock depreciation rate;

$\lambda_{11} = 1, \lambda_{12} = 0.5, \lambda_{21} = 2, \lambda_{22} = 0.2$, representing the labor-investment ratios of sector j of region i ;

$C_{11} = 1, C_{12} = 2, C_{21} = 1, C_{22} = 2$, representing the necessary simple labor of sector j in region i ;

$N_1 = 15, N_2 = 20$ (in thousands), representing the population in regions (in thousands);

$Y_{11}(0) = Y_{12}(0) = Y_{21}(0) = Y_{22}(0) = 0$, representing the income at the beginning of the period;

$I(0) = RMB200$ (in millions).

Suppose the government would adopt $(\varphi_{11}, \varphi_{12}, \varphi_{21}, \varphi_{22}) = (1/4, 1/4, 1/4, 1/4)$ in traditional approach. The optimal cross-region public investment model is resolved with genetic algorithm according to the parameters above. The GA-based solution will show different results.

The chromosomes are selected based on roulette wheel selection approach for next population. Two-point crossover and uniform mutation are performed. And the population is evaluated based on fitness function (17). The results are shown as follows, where \bar{K} is average capital stock, \bar{y} is average income per capita, and Y_{ij} is in *RMB* millions.

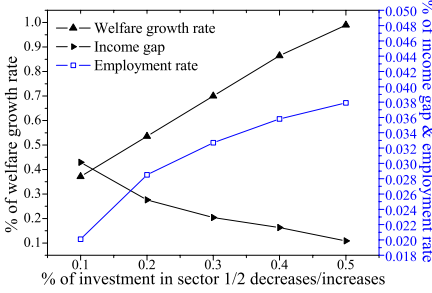
If the investment is divided averagely between regions, solution samples are shown as Table 1. If the investment is divided averagely between sectors, solution samples are shown as Table 2.

Table 1. Investment is divided averagely between regions

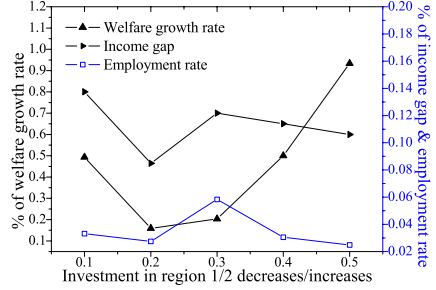
| $(\varphi_{11}, \varphi_{12}, \varphi_{21}, \varphi_{22})$ | $Y_{11}, Y_{12}, Y_{21}, Y_{22}$ | W/K | E/\bar{y} | P |
|--|----------------------------------|-------|-------------|---------|
| $(0, 1/2, 0, 1/2)$ | $(4.94, 10.75, 4.94, 8.59)$ | 2.01% | 42.90% | 37.14% |
| $(1/8, 3/8, 1/8, 3/8)$ | $(8.15, 9.93, 10.11, 8.15)$ | 2.85% | 27.57% | 53.57% |
| $(1/4, 1/4, 1/4, 1/4)$ | $(9.87, 9.03, 12.88, 7.66)$ | 3.27% | 20.37% | 70.00% |
| $(3/8, 1/8, 3/8, 1/8)$ | $(11.62, 7.91, 14.96, 7.12)$ | 3.58% | 16.31% | 86.43% |
| $(1/2, 0, 1/2, 0)$ | $(12.88, 6.51, 16.68, 6.51)$ | 3.79% | 10.86% | 100.00% |

Table 2. Investment is divided averagely between sectors

| $(\varphi_{11}, \varphi_{12}, \varphi_{21}, \varphi_{22})$ | $Y_{11}, Y_{12}, Y_{21}, Y_{22}$ | W/K | E/\bar{y} | P |
|--|----------------------------------|-------|-------------|--------|
| $(0,0,1/2,1/2)$ | $(4.94,6.51,16.68,8.59)$ | 3.32% | 49.35% | 80.00% |
| $(1/8,1/8,3/8,3/8)$ | $(8.15,7.91,10.11,8.15)$ | 2.75% | 15.90% | 46.43% |
| $(1/4,1/4,1/4,1/4)$ | $(9.87,9.03,12.88,7.66)$ | 5.83% | 20.37% | 70.00% |
| $(3/8,3/8,1/8,1/8)$ | $(11.62,9.93,10.11,7.12)$ | 3.05% | 50.05% | 65.00% |
| $(1/2,1/2,0,0)$ | $(12.88,10.75,4.94,6.51)$ | 2.48% | 93.38% | 60.00% |



(a) Investments vary in sectors



(b) Investments vary in regions

Fig. 1. Investments vary in sectors and regions

The results can also be shown in figures more clearly. In Fig. 1(a), the welfare growth rate and employment rate increase while the income gap decreases as the investment to the more labor-intensive sector increases. In Fig. 1(b), the welfare growth rate and employment rate fluctuate but tend to decrease while the income gap fluctuates but tends to increase as the investment to the less-developed region increases. The fluctuations are caused by different labor-investment ratios, simple labor thresholds, etc.

If the government requires $B = 80\%$ and the income gap no higher than 20% , then $(3/8, 1/8, 3/8, 1/8)$ and $(1/2, 0, 1/2, 0)$ are in the feasible solution set. If the government requires $B = 50\%$ and the income gap no higher than 50% , then $(1/8, 3/8, 1/8, 3/8)$, $(1/4, 1/4, 1/4, 1/4)$, $(3/8, 1/8, 3/8, 1/8)$, $(1/2, 0, 1/2, 0)$, and $(0, 0, 1/2, 1/2)$ are in the feasible solution set. The GA-based solution provides scientific foundation for public investment decision, and is much better than the traditional decision rate approach on basis of negotiation.

6 Conclusions

Two new objective functions are integrated into the previous double-objective optimal public investment model. And the complicated multi-objective multi-constraint optimal problem is solved with GA-based solution. Then encoding and decoding approaches are designed according to characteristics of the model. Case study shows the superiority of GA-based solution in providing theoretic foundation for public investment decision on the ground of regional economic growth, equity per capita, and employment rate.

References

1. Chen B.L.: Economic growth with an optimal public spending composition. *Oxford Economic Papers* 58(1), (2006) pp. 123-136
2. Yamano, N., Ohkawara T.: The Regional Allocation of Public Investment: Efficiency or Equity, *Journal of Regional Science*, Vol.40, Issue 2, (2000) pp. 205
3. Smith G.: Creating the conditions for public investment to deliver full employment and environmental sustainability. *International Journal of Environment, Workplace and Employment* Vol.1, No.3/4, (2006) pp. 258-264
4. Tian L., Liu L., Han L., Huang H.: A Genetic Algorithm-Based Double-Objective Multi-constraint Optimal Cross-Region Cross-Sector Public Investment Model. L. Jiao et al. (Eds.): *ICNC 2006, Part II, LNCS 4222*, (2006) pp. 470-479
5. Sakashita, N.: Regional Allocation of Public Investment. Paper of the Regional Science Association, Vol.19, (1967) pp. 161-162
6. Yang X.L.: Improving Portfolio Efficiency: A Genetic Algorithm Approach. *Computational Economics*, Vol.28, Issue 1, (2006) pp. 1-14
7. Gen, M., Cheng, R.W.: *Genetic Algorithms and Engineering Optimization*. John Wiley & Sons, Inc. (2000)
8. Wróblewski J.: Finding Minimal Reducts Using Genetic Algorithm (extended version). P.P. Wang (ed.): *JCIS'95*, (1995) pp. 186-189
9. Hsieh T.Y., Liu H.L.: Genetic Algorithm for Optimization of Infrastructure Investment under Time-Resource Constraints. *Computer-Aided Civil and Infrastructure Engineering*, Vol.19, No.3, (2004) pp. 203-212(10)
10. Metenidis, M. F., Witczak M., Korbicz J.: A Novel Genetic Programming Approach to Nonlinear System Modeling: Application to the DAMADICS Benchmark Problem. *Engineering Applications of Artificial Intelligence*, Vol.17, No.4, (2004) pp. 363-370
11. Pan, Y., Yu, Z.W., Liu, K.J., Dou, W.: A New Multi-Objective Programming Model of QoS-based Multicast Routing Problem. *Computer Engineering and Application*, No. 19 (2003) 155C157
12. Guo, H.Y., Zhang, L., Jiang, J.: Two-Stage Structural Damage Detection Method with Genetic Algorithms. *Journal of Xi'an Jiaotong University*, Vol. 39, No. 5 (2005) 485-489

Many-Objective Particle Swarm Optimization by Gradual Leader Selection

Mario Köppen and Kaori Yoshida

Kyushu Institute of Technology, Dept. Artificial Intelligence,
680-4, Kawazu, Iizuka, Fukuoka 820-8502, Japan
{mkoeppen,kaori}@pluto.ai.kyutech.ac.jp

Abstract. Many-objective optimization refers to multi-objective optimization problems with a number of objectives considerably larger than two or three. This paper contributes to the use of Particle Swarm Optimization (PSO) for the handling of such many-objective optimization problems. Multi-objective PSO approaches typically rely on the employment of a so-called set of leaders that generalizes the global best particle used in the standard PSO algorithm. The exponentially decreasing probability of finding non-dominated points in search spaces with increasing number of objectives poses a problem for the selection from this set of leaders, and renders multi-objective PSOs easily unusable. Gradual Pareto dominance relation can be used to overcome this problem. The approach will be studied by means of the problem to minimize the Euclidean distances to a number of points, where each distance to the points is considered an independent objective. The Pareto set of this problem is the convex closure of the set of points. The conducted experiments demonstrate the usefulness of the proposed approach and also show the higher resemblance of the proposed PSO variation with the standard PSO.

1 Introduction

Recently, there has been growing interest in the application of particle swarm optimization (PSO) to the handling of multi-objective optimization problems. Since the initial presentation of the MOPSO algorithm (Multi-objective Particle Swarm Optimization) [2], a growing number of proposals about corresponding standard PSO variations can be found in the literature. The recent survey of Reyes-Sierra and Coello [10] already classifies nearly thirty of such algorithms. According to [10], the general structure of any such PSO variant can be seen as given in Algorithm 1.1 (also covering the standard “single-objective” PSO). The main difference to a PSO is the notion of “leaders,” which generalizes the common concept of the global best particle in the standard PSO. This regards the fact that in multi-objective optimization, usually, there is not a single optimum but a set of optima solutions. Without the support of an additional, external “decision maker” instance, the problem statement does not entail any further selection criteria that can be applied to this set of optima.

MOPSO and all its successors proved to be competent algorithms to handle the domain of multi-objective optimization, at least for problems posing two or three conflicting objectives. However, no efforts so far have been devoted to the handling of a notably larger number of objectives. More and more, problems with a larger number of objectives are appearing in practice and deserve a deeper study of the question whether they could be handled by the PSO heuristic as well.

Algorithm 1.1. GENERAL PARTICLE SWARM OPTIMIZATION([10](#))

```

Initialize swarm
Locate leader
 $g \leftarrow 0$ 
while  $g < G$ 
  for each particle
    do {
      Update position (Flight)
      Evaluation
      Update lbest
    }
  Update leader
   $g \leftarrow g + 1$ 

```

Notably, approaches to handle these so-called many-objective optimization problems are likely to suffer from the so-called “curse of dimensionality.” In this paper, we will stress on this and identify a weak point in the common scheme for selecting among the leaders, related to the rapidly decreasing probability of finding such leaders in the search space at all. We will propose an approach to overcome this drawback by using a gradual ordering relationship among particles, and compare its performance by means of a scalable many-objective optimization problem that we are going to introduce here as well.

Section 2 will provide the necessary algorithmic concepts and solicit the problem statement. Due to space limitations, multi-objective optimization and PSO will be only briefly touched. The reader is suggested to consider the excellent books of Coello et al. [\[3\]](#) and Deb et al. [\[4\]](#) about multi-objective optimization, and the book of Kennedy and Eberhart to learn about PSO [\[6\]](#). The used ranking by Fuzzy Pareto Dominance (FPD), which is employed in the PSO generalization, will be recalled in Sect. 2 as well. The so-called P* many-objective optimization problem will be introduced in Sect. 3. Section 4 then provides the results of test runs of multi-objective PSOs on this problem, and a discussion of the results. The concluding section and the references will be at the end of this paper.

2 Multi-objective Particle Swarm Optimization

In multi-objective optimization, the mapping of a feature vector space F into an objective vector space O is considered, where points of F are sought, having all their objective values as small as possible. For comparing two points in objective space, the common notion of Pareto dominance (or just dominance) is employed.

Given two vectors $x = (x_1, x_2, \dots, x_n)$ and $y = (y_1, y_2, \dots, y_n)$ of same size, it is said that vector x “Pareto dominates” y , if each component of x is smaller than or equal to the corresponding component of y , and at least one component is smaller: $x <_D y \leftrightarrow \forall i(x_i \leq y_i) \wedge \exists k(x_k < y_k)$. A similar definition for the maximum case, or the case “smaller/larger-or-equal” (so-called weak dominance) can be provided as well.

For simplicity, we also consider the dominance relation among points in the feature space, if their corresponding objective vectors are in such a relation. The set of all objective vectors assigned to feasible feature vectors, which are not dominated by any other objective vector is the so-called “Pareto front” of the multi-objective optimization problem. The set of the corresponding feature vectors will be referenced as Pareto set in the following.

The general task of an multi-objective optimization algorithm is to find a representation of the Pareto front of the given mapping. A new class of evolutionary algorithms, with specialized selection operators to regard for the multiple objectives, has gained much attractiveness for the handling of such problems. As already said in the introduction section, this interest has expanded to the Particle Swarm Optimization algorithm in between.

Here, we consider the multi-objective PSO (MOPSO) that was recently proposed by Alvarez et al. [11], see the algorithm listing 2.1. It shows the same global structure as the general PSO given in listing 1.1. The procedure to select the leader is based on selecting from an archive that stores all non-dominated positions found by the algorithm during its course so far.

Algorithm 2.1. MOPSO_{rand}([11])

```

A ← ∅
Initialize particles
for g ← 1 to G
do {
  for n ← 1 to N
  do {
    for k ← 1 to K
    do {
      vnk = wvnk + r1(Pnk - xnk) + r2(Gnk - xnk)
      xnk = xnk + vnk + ε
      xn ← enforceConstraints(xn)
      on ← f(xn)
    }
    if ~ ∃u ∈ A : u ≤D xn
    then {
      A ← {u ∈ A | xn <D u}
      A ← A ∪ xn
    }
    if xn ≤D Pn ∨ (xn <D Pn ∧ Pn <D xn)
    then Pn ← xn
    Gn ← selectGuide(xn, A)
  }
}

```

The parameters used in the listing are: x_n and v_n are position and velocity of the n -th particle, A stands for the archive, G is the maximum number of cycles, N is the number of particles, K the dimension of a particles’ vector, P_n is the local best position of particle x_n , and G_n the leader (or guide) for this particle, used to update its position according to the general swarm equations. The parameters r_1 and r_2 are random numbers drawn from the interval $[0, \dots, w_{local}]$ and $[0, \dots, w_{global}]$ respectively, w is the weight of former

velocity, and ϵ so-called curl parameter, a small random number. The procedure *enforceConstraints()* ensures the swarm to stay within the feasible space, and the procedure *selectGuide()* selects the leader for each particle, and will be detailed in a moment.

2.1 Choices for the Leader Selection

In [1], three procedures for the selection of the leader from the archive were proposed. All of them assume the presence of particles being dominated by vectors that are already stored in the archive, or being dominated by new positions. The procedure *RAND* randomly selects one of the archive members dominating particle x_n as leader for this particle. If there is no such particle, an archive member is randomly selected.

However, for a larger number of objectives, the second choice becomes more likely. Equation (1), taken from [7], gives the expected size of the Pareto front of m points randomly selected from the n -dimensional unit hypercube

$$e_m(n) = m - \sum_{k=1}^m \frac{(-1)^{k+1}}{k^{n-1}} \binom{m}{k}. \quad (1)$$

A closer look on this equation gives that the probability of finding two points with one dominating the other drops exponentially with the problem dimension. For example, for 15 objectives and 10 particles, the probability to have a randomly selected dominated point is already as low as 0.0027.

For this reason, we are considering gradual Pareto dominance here, as it was presented in [9]. Given a set of points, a “ranking value” is assigned to each point a . This ranking value is the maximal value of the “degree of being dominated” by any other element of the set. The dominance degree is computed as

$$\mu_{pmin}(x, y) = \prod_i \left[\frac{x_i}{y_i} \right], \quad (2)$$

where the notion of a bounded division was used:

$$\left[\frac{x}{y} \right] = \begin{cases} 1, & \text{if } y \leq x \\ x/y, & \text{if } x < y \end{cases}. \quad (3)$$

The ranking value of a dominated point is 1, otherwise its between 0 and 1. Points with smaller ranking values can be considered to be less dominated by the other points in the set. The ranking value is scale independent, and points close to other points in the set yields higher ranking values than points that are more distant. Thus, the ranking values also punish crowding of points at the same location. Finally, the ranking value is set-dependent. If in a set a point x has a lower ranking value than y , adding an element to the set close to x may reverse the ranking value size relation.

In this paper, the procedure *FPD* for *selectGuide()* selects as leader for all particles the particle with the lowest ranking value within the set of all swarm particles.

3 The P^* Many-Objective Optimization Problem

The evolutionary multi-objective optimization community maintains a set of benchmark measures for the performance assesment of algorithms, with the DTLZ suite of problems [5] being among the most popular. Unfortunately, not much is known about these problems for the many-objective case. This circumstance has already been remarked with the introduction of the Pareto Box problem, which identifies objective vector and feature vector [8]. However, issues of vector components becoming 0 and the bounded domain of positive vector components hardens the use of the Pareto Box problems, especially for swarms.

Here, we are introducing a related problem, referred to as P^* problem for indicating the variable number of points from which the objectives are derived.

Given is a set P of m points P_i in the Euclidian plane (the case of two dimensional Euclidian space is completely sufficient for the present analysis). The feature space F equals the Euclidian plane, where the points P_i are located. The objective space O is an m -dimensional vector space. For a given point x in the feature space, its objective vector $o(x)$ is the vector with the components $o_i = d(x, P_i)$ for $i = 1$ to m , where $d(x, y)$ is the Euclidian distance of two points $x, y \in F$. Thus, the objectives to minimize are the distances to a given collection of points, where the distance to any of these point is treated as an independent objective.

The Pareto set of this problem equals the convex closure of the points P_i . To see this, consider Fig. 1. The left subfigure shows that for any point x outside the convex closure there is at least one point y that is closer to all points of P . In the subfigure, the line $\overline{A_i A_{i+1}}$ represents one segment of the convex hull. All points on this line or on the other side of this line than x are more close to y than to x .

To see that none of the points of the convex closure dominates any other, consider the right subfigure of Fig. 1. By connecting any two points U and V of the convex closure and drawing the perpendiculars to this line trough U and through V , the convex closure is segmented into three parts. There is at least one point of the point set A (and thus of P) located to the l.h.s. of the perpendicular

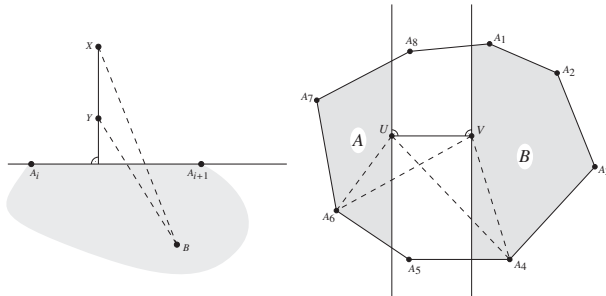


Fig. 1. Proof that the Pareto set of the P^* problem for some points A_i is the convex closure of these points

through U (indicated by encircled A in the figure), or located on this line, and there is at least one point of A belonging to the r.h.s. of the perpendicular through V or on it (indicated by encircled B). Otherwise, the shape would not be convex. Now, point U is more close to any point of A than V , and point V is more close to any point of B than U . Neither U nor V can dominate the other.

Having thus a rather simple solution structure, the problem is worth a study for a heuristic algorithm for several reasons:

- the number of objectives can be easily scaled
- by reducing the area enclosed by the convex closure, the effort for random search (the “Monte-Carlo Barrier”) can be easily increased
- typical performance measures (as average distance to Pareto front, number of particles belonging to the Pareto front) can be directly computed
- as the feature space is two-dimensional, the results can be directly visualized; however, extension to higher-dimensional spaces is straightforward
- the search space is not bounded
- the problem is a continuous optimization problem
- boundary conditions can be directly included
- crowding in objective space directly corresponds to crowding in feature space
- modelling of algorithm behaviour seems feasible
- by using the distance to the center of gravity of the points instead, a comparison to the single-objective case becomes possible

In the following, we are considering the performance of three algorithms on the P^* problem.

4 Results and Discussion

Several experiments have been conducted to study the behaviour of three algorithms on the P^* problem. The three algorithms were the MOPSO of Alvarez et al. using *RAND* (algorithm $MOPSO_{rand}$), the proposed usage of *FPD* (algorithm $MOPSO_{fpd}$) as procedures for *selectGuide()*, and the standard PSO (algorithm PSO) by taking the distance to the c.o.g. of the points P as objective. The result that we want to present here was achieved with the following settings: swarm sizes were 10 particles each; the swarms were randomly initialized around point 0, with deviation of 0.1 and max initial velocity of 0.001; the weight of the global best was 0.08, the weight of the local best 0.02, the weight of the former velocity 0.985. For each target problem, the average distance of the c.o.g. of the particle positions after 1000 cycles for 20 different random initializations was computed. The target problems were given by placing a set of 15 circular points at a radius of 0.01 around the center $(i/10, i/10)$ with i going from 1 to 20.

The results can be seen in Fig. 2. For reference, the initial distance of the target to the starting point of the swarm has been plotted as well. The most notable fact is the nearly complete failure of the algorithm $MOPSO_{rand}$ to find the target set, once the target gets placed beyond the $(0.8, 0.8)$ offset. The constant bias of about 0.1 of $MOPSO_{rand}$ performance to the reference line refers to the

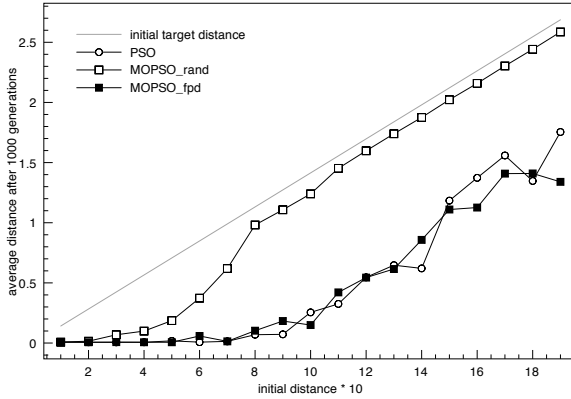


Fig. 2. Plots of the average distances to the target after 1000 cycles vs. different initial target distances for a problem with 15 objectives

fact that the initial positions varied by about 0.1 around the point 0. Before the failure, it has to be noted that the averages for $MOPSO_{rand}$ are taken from more or less binary cases: the $MOPSO_{rand}$ swarm was either reaching the target set, or got stuck at the initial position.

For understanding this failure, consider a snapshot of the $MOPSO_{rand}$ swarm particles taken anywhere in such a situation (see Fig. 3). The snapshot also shows the positions of the local best, as they are kept by each individual, and the target points. It can be seen that in such a situation, the local best positions establish a kind of “trap” for the swarm. The swarm is surrounded by these positions, and the probability of finding a closely position that either dominates the local best positions, or can get added to the archive is nearly zero. The archive so far is not able to guide the swarm out of this trap. So, the swarm stays within the area surrounded by the local best positions, and the local best positions never get updated by new dominating positions. That’s the reasoning, given in humble words. Any manner of quantifying the situation of a MOPSO getting stuck will only provide additional evidence for the fact that this is a ubiquitous feature of all algorithms that depends on the finding of dominated points, to perform their operations.

The algorithm $MOPSO_{fpd}$ shows a much more improved performance, as it is capable to approach the target even if being initially placed distant from the target set. However, naturally the explorational effort for $MOPSO_{fpd}$ is also increasing. But it has to be taken into account that the average values shown for the $MOPSO_{fpd}$ were taken from a variety of distance values: in nearly no case, the $MOPSO_{fpd}$ swarm got ever stuck at the initial position, as it happened many times for the $MOPSO_{rand}$. It could always escape the trap set up by the local best, even if no dominating position was found, as it is also rewarding nearly dominating positions. By extending the number of cycles, the $MOPSO_{fpd}$ swarm may still approach the target.

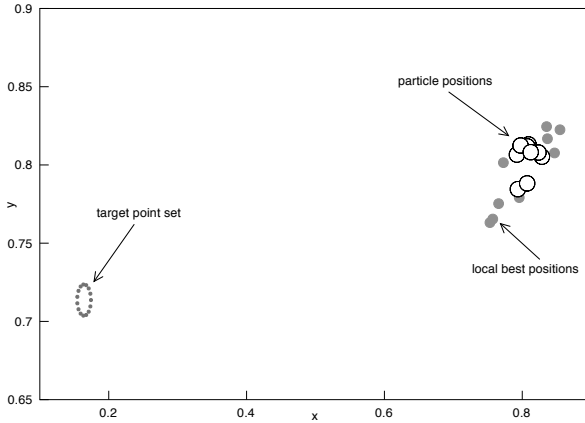


Fig. 3. A situation where a $MOPSO_{rand}$ swarm is trapped within the local best positions of the particles

The performance measure for the standard PSO has been taken as well. It is remarkable how strongly $MOPSO_{fpd}$ performance resembles the one of the standard PSO, as both plots go nearly equal. This encourages the choice of the FPD-based leader selection scheme as a way to expand the standard PSO to the many-objective optimization domain.

5 Conclusions

In this paper, the extension of Multi-objective Particle Swarm Optimization (MOPSO) to the case of many-objective optimization has been studied. Here, “many-objective” stands for a number of objectives considerably larger than two or three. It has been demonstrated how algorithms that rely on the presence of dominated points may get stuck in their search for the Pareto front. The notion of Pareto dominance is not fully suited to the case of many objectives. Pareto dominance requires ALL components of a vector to be smaller than the corresponding components of the other vector. This becomes more and more unlikely, as the number of components increases. As a consequence, such an MOPSO algorithm may get trapped by the local best selection of the particles itself, without being able to find new dominating positions. As a countermeasure, this paper presented the use of gradual Pareto dominance, to fuse the relative amount of smaller vector components, and the degree by which they are smaller, into a single measure. Doing leader selection based on these so-called “ranking values” allows for the design of a MOPSO, which better resembles a standard single-objective PSO in the multi-objective case.

Acknowledgments. A researcher involved in this study has been supported by a JSPS grant.

References

1. Julio E. Alvarez-Benitez, Richard M. Everson, and Jonathan E. Fieldsend. A MOPSO Algorithm Based Exclusively on Pareto Dominance Concepts. In Carlos A. Coello Coello, Arturo Hernández Aguirre, and Eckart Zitzler, editors, *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*, pages 459–473, Guanajuato, México, March 2005. Springer. Lecture Notes in Computer Science Vol. 3410.
2. Carlos A. Coello Coello and Maximino Salazar Lechuga. MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization. In *Congress on Evolutionary Computation (CEC'2002)*, volume 2, pages 1051–1056, Piscataway, New Jersey, May 2002. IEEE Service Center.
3. Carlos A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, New York, May 2002. ISBN 0-3064-6762-3.
4. Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.
5. Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable Multi-Objective Optimization Test Problems. In *Congress on Evolutionary Computation (CEC'2002)*, volume 1, pages 825–830, Piscataway, New Jersey, May 2002. IEEE Service Center.
6. James Kennedy and Russell C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann Publishers, San Francisco, California, 2001.
7. Mario Köppen, Raul Vicente Garcia, and Bertram Nickolay. Fuzzy-pareto-dominance and its application in evolutionary multi-objective optimization. In *Evolutionary Multi-Criterion Optimization, Third International Conference, EMO 2005, Guanajuato, Mexico, March 9-11, 2005. Proceedings*, LNCS 3410, pages 399–412. Springer Berlin / Heidelberg, 2005.
8. Mario Köppen, Raul Vicente Garcia, and Bertram Nickolay. The pareto-box problem for the modelling of evolutionary multi-objective optimization. In *Adaptive and Natural Computing Algorithms. Proceedings of the ICANNGA 2005, Coimbra, Portugal*, pages 194–197, 2005.
9. Mario Köppen and Raul Vicente Garcia. A fuzzy scheme for the ranking of multi-variate data and its application. In *Proceedings of the 2004 Annual Meeting of the NAFIPS (CD-ROM)*, pages 140–145, Banff, Alberta, Canada, 2004. NAFIPS.
10. Margarita Reyes Sierra and Carlos A. Coello Coello. Multi-objective particle swarm optimizers: A survey of the state-of-the-art. *International Journal of Computational Intelligence Research*, 2(3):287–308, 2006.

Mixed Ant Colony Optimization for the Unit Commitment Problem

Ana-Talida Serban^{1,2} and Guillaume Sandou¹

¹Supélec, Automatic Control Department
3, rue Joliot Curie, F-91192 Gif-sur-Yvette, France
Guillaume.Sandou@supelec.fr

²Politehnica University of Bucharest
Faculty of Automatic Control & Computer Science
060042-Bucharest, Romania

Abstract. In this paper, a mixed integer programming method based on ant colony optimization is presented, and applied to the classical Unit Commitment problem. The idea is to reformulate the problem into a graph exploration structure, and to use discrete ant colony optimization to explicitly take into account time down, time up and demand constraints in the optimization procedure. This method is coupled with a continuous ant colony algorithm to compute produced powers. Results, obtained on relatively small cases, show the viability of the proposed approach: a near optimal solution, with guarantees of feasibility, can be computed with low computation times.

1 Introduction

Unit Commitment is a classical mixed integer problem in power systems, which aims to compute the optimal scheduling of several production units while satisfying consumer's demand and technical constraints. On/off variables, which are integer variables, and produced powers, which are real ones are the optimization variables. Numerous methods have already been applied to solve this classical optimisation problem. They are for example listed in [1] and are here briefly called up.

Firstly, exact solution methods have been tested: exhaustive enumeration, Branch and Bound [2], dynamic programming [3]. These methods suffer from combinatorial complexity, and thus, efficient approximated methods are required. Furthermore, taking into account temporal constraints, such as minimum time up or time down constraints, may be difficult. It is the case for dynamic programming solution, for which extended spaces have to be considered.

Deterministic methods can be used such as priority lists [4]. Due to numerous constraints, they are strongly suboptimal. Constraints are considered by Lagrangian relaxation [5]. Constraints which couple several production units are relaxed, and the Unit Commitment problem can be divided into several optimization problems (one

per production unit). Solutions are computed with dual problems. The non convexity of the objective function implies a duality gap and no guarantee can be given on the actual optimality. An iterative procedure has to be performed: solution of the optimization problems (fixed Lagrange multipliers) and update of the multipliers. The update can be made with genetic algorithms [6] or by subgradient methods [7].

Metaheuristics methods have also been intensively used (see [1]). A random but relevant enumeration of variables is performed. There is no guarantee on the actual optimality, but a very suitable solution with low computation times can be found. A simulated annealing approach is used in [8], tabu search is used in [9] and genetic algorithms are used in [10]. Cooperative algorithms are also developed to combine the advantages of several methods, like genetic algorithms and simulated annealing in [11]. The management of the feasibility of solutions may be difficult with such methods. The algorithm “moves” randomly in the search space, and so, there is no guarantee that the final solution is in the feasible set. This is the case for Unit Commitment, as the feasible set is much smaller than the search space.

In this paper a mixed ant colony optimization method is used to compute solutions for the Unit Commitment problem. Indeed, the use of a constructive algorithm allows explicitly handling the constraints and so, the guarantee of feasibility is achieved. Ant colony has already been used in Unit Commitment literature in [12], [13], [14] and [15]. For example, in [12], real variables are computed using an extension of discrete ant colony to continuous spaces as depicted in [16]. In [13], real variables are computed using a lagrange multipliers method. In [15], they are computed with a lambda iteration method. In this article, the idea is to use the structure of the problem to solve the problem with a purely ant colony algorithm: a discrete ant colony algorithm computes the binary variables, taking into account the time down, time up and demand constraints. Simultaneously, a continuous ant colony algorithm performs the computation of produced powers (real variables).

The paper is organized as follows. First, the Unit Commitment problem is called up in section 2. The mixed ant colony algorithm is depicted in section 3; it is based on a graph exploration reformulation. Section 4 presents numerical results about Unit Commitment solution. Finally, concluding remarks are drawn in section 5.

2 Unit Commitment Problem

Unit Commitment refers to the optimal scheduling of several production units:

$$\min_{\{u_n^k, Q_n^k\}} \sum_{n=1}^N \left(\sum_{k=1}^K (c_{prod}^k (Q_n^k, u_n^k) + c_{on/off}^k (u_n^k, u_{n-1}^k)) \right), \tag{1}$$

where N is the length of time horizon, K is the number of production unit, u_n^k (resp. Q_n^k) is the on/off status (resp. produced power) of the production unit k during the time interval n . Production costs c_{prod}^k and start up and shut down costs $c_{on/off}^k$ can be defined by these equations:

$$\begin{cases} c_{prod}^k(Q_n^k, u_n^k) = a_2^k(Q_n^k)^2 + a_1^k Q_n^k + a_0^k u_n^k \\ c_{on/off}^k(u_n^k, u_{n-1}^k) = c_{on}^k u_n^k (1 - u_{n-1}^k) + c_{off}^k u_{n-1}^k (1 - u_n^k). \end{cases} \tag{2}$$

where $a_k^0, a_k^1, a_k^2, c_{on}^k, c_{off}^k$ are technical data of production unit k .
 The constraints of the problem are:

- capacity constraints

$$Q_{min}^k u_n^k \leq Q_n^k \leq Q_{max}^k u_n^k, \quad \forall n \in \{1, \dots, N\}, \forall k \in \{1, \dots, K\}, \tag{3}$$

- consumer demand fulfilling

$$\sum_{k=1}^K Q_n^k \geq Q_n^{dem}, \quad \forall n \in \{1, \dots, N\}, \tag{4}$$

- time up and time down constraints

$$\begin{cases} (u_{n-1}^k = 0, u_n^k = 1) \Rightarrow (u_{n+1}^k = 1, u_{n+2}^k = 1, \dots, u_{n+T_{up}^k-1}^k = 1) \\ (u_{n-1}^k = 1, u_n^k = 0) \Rightarrow (u_{n+1}^k = 0, u_{n+2}^k = 0, \dots, u_{n+T_{down}^k-1}^k = 0) \end{cases} \tag{5}$$

3 Mixed Ant Colony Optimization Formulation

Ant Colony Optimization was firstly introduced by Marco Dorigo. This method is based on the way natural ants are searching for food. The ant colony is able to find the shortest path from the nest to the food, despite the fact that a single ant does not have a global vision of the problem. Ants move in the search space and influence each other with the help of the pheromone trail. For a more precise description, see [17]. This behavior can be exploited to design efficient stochastic algorithms for graph exploration problems, such as the classical traveling salesman problem in [18].

3.1 Graph Exploration Formulation

The Unit Commitment problem can be formulated as a graph exploration problem as shown in figure 1, as in previous work [19]. The nodes of the graph represent all the possible states (u_n^1, \dots, u_n^K) of production system, for all the time intervals. The aim is to go from one of the possible states at time 1, to one of the possible states at time N , while satisfying all the constraints and minimising global costs defined in equation (1). For each edge $(u_n^1, \dots, u_n^K) \rightarrow (u_{n+1}^1, \dots, u_{n+1}^K)$ of the graph, start-up and shut-down costs are added. Production costs are also associated to nodes.

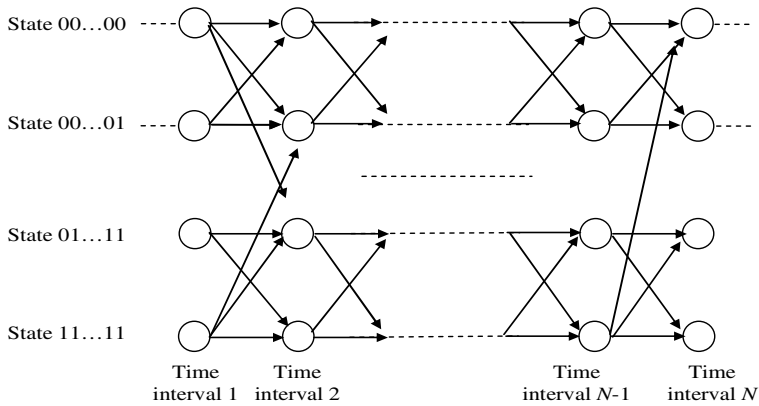


Fig. 1. Graph structure for Unit Commitment

3.2 Computation of Binary Variables

As in the classical ant colony algorithm, during iteration t of the algorithm, F ants walk on this graph. When an ant f is in state $i = (u_n^1, \dots, u_n^K)$, the probability of choosing the next state $j = (u_{n+1}^1, \dots, u_{n+1}^K)$ is defined by the probabilistic law:

$$p_i^{(f)}(j) = \frac{\eta_{ij}^\alpha \tau_{ij}(t)^\beta}{\sum_{m \in J_f(i)} \eta_{im}^\alpha \tau_{im}(t)^\beta} \tag{6}$$

- $\tau_{ij}(t)$ is the pheromone trail on edge $i = (u_n^1, \dots, u_n^K) \rightarrow j = (u_{n+1}^1, \dots, u_{n+1}^K)$ during iteration t . Its value depends on the results of previous ants, and will be discussed in section 3.4.
- η_{ij} is called the attractiveness. This parameter refers to the « local choice ». In the travelling salesman problem, η_{ij} is the inverse of the distance between town i and town j . For Unit Commitment, the following remark can be made: when next node has to be chosen, the best local candidate is the node for which the gap between the maximum produced power and the predicted demand is the smallest. This is the basis of the definition of attractiveness for Unit Commitment; for more details, see previous work [19].
- α and β are weighting factors.
- $J_f(i)$ is the feasible set. This feasible set contains a priori all 2^K states. But, those states which do not satisfy time up and time down constraints, and those states which do not satisfy consumers' demands, are to be removed. Note that, even if produced powers are not known yet, it is possible to check the possibility of consumer's demand satisfaction with the equation:

$$\sum_{k=1}^K Q_{\max}^k u_n^k \geq Q_n^{dem} \tag{7}$$

Finally, $J_f(i)$ sets are recursively constructed for each ant, and lead to the guarantee of the feasibility of solutions. In this study, the principles of “Max-Min Ant System”,

proposed in [20], are used. The pheromone trail for each edge is bounded to $[\tau_{\min}, \tau_{\max}]$. The goal of this limitation is to avoid premature convergence of the algorithm to a local minimum: all the edges, even if they seem to be unprofitable, can be chosen by successive ants. After the ant has completed its path, it is possible to evaluate the solution by solving the real optimization problem defined in equation (1), with fixed binary variables. However, as this kind of problem would have to be solved numerous times, this is too much time consuming. That is why a continuous ant colony method is added to this method to compute real variables from integer ones.

3.3 Computation of Real Variables

For each integer solution $U = \{u_n^k; n = 1, \dots, N; k = 1, \dots, K\}$, computed with discrete ant colony algorithm, a real solution $Q = \{Q_n^k; n = 1, \dots, N; k = 1, \dots, K\} = \{x^1, \dots, x^{KN}\}$ has to be associated. To compute these real variables, a continuous ant colony optimization algorithm defined by Socha and Dorigo in [21] is used and is here called up. A matrix **T** of s real solutions, called archive matrix of solutions, is stored and represented in figure 2.

$$\mathbf{T} = \begin{bmatrix} x_1^1 & x_1^2 & \dots & x_1^i & \dots & x_1^{KN} \\ x_2^1 & x_2^2 & \dots & x_2^i & \dots & x_2^{KN} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_j^1 & x_j^2 & \dots & x_j^i & \dots & x_j^{KN} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_s^1 & x_s^2 & \dots & x_s^i & \dots & x_s^{KN} \end{bmatrix} \quad \mathbf{H} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_j \\ \vdots \\ f_s \end{bmatrix} \quad \mathbf{\Omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_j \\ \vdots \\ \omega_s \end{bmatrix} \quad \mathbf{P} = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_j \\ \vdots \\ p_s \end{bmatrix}$$

Fig. 2. Archive matrix for the continuous ant colony algorithm

These global solutions (binary and real variables) have been evaluated with respect to the objective function defined in equation (1) and their costs are stored in **H** (see figure 2), whose components are then:

$$f_j = f(U_j, Q_j) = \sum_{n=1}^N \left(\sum_{k=1}^K (c_{prod}^k(Q_n^k, u_n^k) + c_{on/off}^k(u_n^k, u_{n-1}^k)) \right). \tag{8}$$

The stored solutions are sorted according to their costs:

$$f_1 \leq f_2 \leq \dots \leq f_r \leq \dots \leq f_s. \tag{9}$$

Weights are defined according to the ranks of the solutions in the matrix. For the solution with rank r , the weight is:

$$\omega_r = \frac{1}{qs\sqrt{2\pi}} e^{-\frac{(r-1)^2}{2q^2s^2}}. \tag{10}$$

The coefficient q is a parameter of the algorithm. Finally, a discrete probability distribution is defined from these weights:

$$p_r = \frac{\omega_r}{\sum_{j=1}^s \omega_j}. \quad (11)$$

To compute a new real solution, a “model ant”, say l , is chosen, according to this discrete probability distribution. Each real variable x_{new}^i , $i = 1, \dots, KN$, is randomly chosen with a Gaussian probability distribution whose mean and standard deviation is computed by:

$$\begin{cases} \mu_{new}^i = x_l^i \\ \sigma_{new}^i = \frac{\xi}{s-1} \sum_{m=1}^s |x_m^i - x_l^i| \end{cases}. \quad (12)$$

The coefficient ξ is also a parameter of the algorithm. Remember that these real solutions are computed from feasible binary solutions. Therefore, equation (7) is satisfied. But, when real variables are randomly chosen, consumer’s demands may not be fulfilled. Furthermore, the random selection of produced powers may lead to overproduction, which is highly unprofitable. To prevent these problems, the following procedure is applied:

- Select produced powers Q_n^k with the previous algorithm,
- If $\sum_{k=1}^K Q_n^k u_n^k > Q_n^{dem}$ (resp. $\sum_{k=1}^K Q_n^k u_n^k < Q_n^{dem}$), then choose randomly one of switched on units, and decrease (resp. increase) the corresponding produced power until $\sum_{k=1}^K Q_n^k u_n^k = Q_n^{dem}$. If it is not sufficient, choose several production units. Note that it is always possible to do so, as equation (7) is satisfied by binary variables.

3.4 Pheromone Updating and Evaporation

The algorithm is based on a positive feedback, as current ants try to influence future ants by enforcing the pheromone trail. As a result, to avoid premature convergence, it is necessary to forget the past mistakes. This is done by the pheromone evaporation. At the end of iteration t of the algorithm, F mixed ants have computed F feasible solutions for the initial problem. For binary variables, the pheromone trail is updated, with the following equation, for each edge of the graph of figure 1:

$$\tau_{ij}(t+1) = (1 - \rho) \tau_{ij}(t) + \Delta \tau_{ij}(t) \quad (13)$$

ρ is the evaporation coefficient. This coefficient can be viewed as an analogy with physical evaporation of pheromone in nature. $\Delta \tau_{ij}$ is the updating coefficient. Its value depends on the results of ants in iteration t . In classical “Max-Min Ant System”, see [20], the update is performed using three mechanisms: iteration best, global best and restart-best. In this study, only iteration-best procedure is used: each ant is evaluated with respect to the objective function (equation (1)). An elitism algorithm is used: only the best ant of current iteration is allowed to lay some pheromone on each edge it

has used. However, if the best ant at current iteration has performed the same solution as the one computed at the previous iteration, no pheromone is added again on the corresponding path. This can be viewed as a way to reinforce the exploration of the search space, or local restart mechanism. At present, no global-best update procedure (the best solution found from the beginning on the algorithm adds pheromone) has been used: it has been observed that it may lead to premature convergence of the algorithm. However, adaptive strategies are among forthcoming works of this study.

For real variables, the solutions computed by best ants are stored in matrix \mathbf{T} , replacing old solutions whose corresponding costs were too bad. Once again, the past is slowly forgotten.

4 Numerical Results

The algorithm has been implemented with Matlab 6.5 and a Pentium IV, 2 GHz. To test the viability of the approach, it has been tested on relative small cases (4 unit benchmark case). Characteristics are given in table 1. The time horizon is 24 hours, with a sampling time of one hour. Thus, the optimization problem is made of 96 binary variables, and 96 real variables. For this benchmark example, linear costs have been considered. Thus, it is still tractable to compute the global optimum, for example with a “Branch and Bound” method, so as to evaluate the gap to optimality. The optimum, computed with this method, is 8780 €.

Table 1. Characteristics of the benchmark example

| Unit | Q_{\min} (MW) | Q_{\max} (MW) | a_0 (€) | a_1 (€/MWh) | c_{on} (€) | c_{off} (€) | T_{down} (h) | T_{up} (h) |
|------|--------------------|--------------------|--------------|------------------|------------------------|-------------------------|--------------------------|------------------------|
| 1 | 10 | 40 | 25 | 2.6 | 10 | 2 | 2 | 4 |
| 2 | 10 | 40 | 25 | 7.9 | 10 | 2 | 2 | 4 |
| 3 | 10 | 40 | 25 | 13.1 | 10 | 2 | 3 | 3 |
| 4 | 10 | 40 | 25 | 18.3 | 10 | 2 | 3 | 3 |

As the algorithm is stochastic, statistical results are provided to evaluate the behavior of the algorithm. For the tests, the algorithm is performed 100 times. Mean and best cases are reported as well as the standard deviation of the results. Results are given in table 2, for 100 and 200 iterations of the algorithm. Several tests have been performed so as to get suitable values for parameters. Finally, satisfying values are as follows:

- $\alpha = 1; \beta = 2$
- $\rho = 0.2,$
- $q = 1; \xi = 0.8,$
- $F = 20, s = 100.$

An iteration of the algorithm is about 0.5 second time consuming. Results show that the method gives promising results. The quality of the method may not be as good as other dedicated methods, but it is very tractable and gives quickly medium

quality solutions, with guarantees of feasibility. Research is done to improve the quality of the method by choosing suitable coefficients and hybridize the method with other metaheuristics such as genetic algorithms and local search methods. The idea would be to use the method as a “fast feasible solution generator”. Promising results about cooperative optimization methods ant colony/genetic algorithm have been obtained in previous works [22] for a purely integer programming formulation of Unit Commitment.

Table 2. Optimization results for the benchmark example

| | Best Case | Mean | Standard deviation |
|------------------------|-----------------------------------|------------------------------------|---------------------------|
| After 100 iter. | 9.18 10 ³ € (+4.5%) | 9.78 10 ³ € (+11.4%) | 451 € |
| After 200 iter. | 8.98 10 ³ € (+2.3%) | 9.48 10 ³ € (+7.7%) | 370 € |

5 Conclusion and Forthcoming Works

A mixed ant colony optimization algorithm has been presented in this paper. The idea is to use a purely ant colony algorithm both for integer and real variables to solve the Unit Commitment problem. Thus, a coupling of a discrete ant colony algorithm and a continuous one has been developed. Finally, as this algorithm is a constructive one, the satisfaction of all constraints is guaranteed by the whole procedure. The advantage of this method is that penalty functions are not required to solve the problem: the guarantee of the solution feasibility does not lead to the tuning of new extra parameters. Finally, numerical results obtained with a benchmark example show that the method is a promising one, as interesting solutions can be computed with very low computation times.

At present, several interesting questions are investigated. Among them is the extension of the method to more generic mixed optimization problems. Indeed, the developed approach is well suited to Unit Commitment problem as all constraints can be taken into account by considering only binary variables, which is of course not always the case for mixed problems. Forthcoming works deals also with the hybridization of the proposed method with other classical optimization methods such as genetic algorithms or local search algorithms, the extension to larger scale cases of Unit Commitment, and the extension of the algorithm to other classical optimization problems of the literature.

References

1. Padhy N. P., Unit commitment – a bibliographical survey, IEEE Transactions on Power Systems, Vol. 9 (2004), pp.1196-1205.
2. Chen C.-L, Wang S.-C., Branch and Bound scheduling for thermal generating units, IEEE Transactions on Energy Conversion, Vol. 8, n°2 (1993), pp. 184-189.

3. Ouyang Z., Shahidehpour S. M., An intelligent dynamic programming for unit commitment application, *IEEE Transactions on Power Systems*, Vol. 6, n°3 (1991), pp. 1203-1209.
4. Senjyu T., Shimabukuro, K., Uezato K., Funabashi T., A fast technique for Unit Commitment problem by extended priority list, *IEEE Transactions on Power Systems*, Vol. 19, n° 4 (2004), pp. 2119-2120.
5. Zhai Q; Guan X., Unit Commitment with identical units: successive subproblems solving method based on Lagrangian relaxation, *IEEE Transactions on Power Systems*, Vol. 17, n°4 (2002), pp. 1250-1257.
6. Cheng C.-P., Liu C.-W., Liu C.-C. Unit Commitment by Lagrangian Relaxation and Genetic Algorithms, *IEEE Transactions on Power Systems*, Vol. 15, n° 2 (2000), pp. 707-714.
7. Dotzauer E., Holmström K., Ravn H. F., Optimal Unit Commitment and Economic Dispatch of Cogeneration Systems with a Storage, 13th Power Systems Computation Conference, Trondheim, Norway (1999), pp. 738-744.
8. Yin Wa Wong S., An Enhanced Simulated Annealing Approach to Unit Commitment, *Electrical Power & Energy Systems*, Vol. 20, n° 5 (1998), 359-368.
9. Rajan C. C. A, Mohan M. R. An evolutionary programming-based tabu search method for solving the unit commitment problem, *IEEE Transactions on Power Systems*, Vol. 19, n°1 (2004), pp. 577-585.
10. Swarup K . S., Yamashiro, Unit commitment solution methodology using genetic algorithm, *IEEE Transactions on Power Systems*, Vol. 17, n°1 (2002), pp. 87-91.
11. Cheng C.-P., Liu C.-W., Liu C.-C., Unit Commitment by annealing-genetic algorithm, *Electrical Power and Energy Systems*, Vol. 24 (2002), pp. 149-158.
12. Song Y. H., Chou C. S., Stonham T. J., Combined heat and power economic dispatch by improved ant colony search algorithm, *Electric Power System Research*, Vol. 52 (1999), pp. 115-121.
13. Simon S. P., Padhy N. P., Anand R. S., An ant colony system approach for unit commitment problem, *Electrical Power & Energy Systems*, Vol. 28, n° 5 (2006), 315-323.
14. Sisworahardjo N. S., El-Keib A. A., Unit commitment using the ant colony search algorithm, *Large Engineering Systems Conference on Power Engineering*, Halifax, Canada (2002).
15. Sum-Im T., Ongsakul W., Ant colony search algorithm for unit commitment, *IEEE International Conference on Industrial Technology*, Maribor, Slovenia (2003).
16. Bilchev G., Parmee I., The ant colony metaphor for searching continuous spaces, *Lecture Notes in Computer Science*, Vol. 993 (1995), pp. 25-39.
17. Dorigo M., Maniezzo V., Colomi A., The Ant System: Optimization by a Colony of Cooperating Agents, *IEEE Transactions on Systems, Man and Cybernetics-Part B*, Vol. 26, n° 1 (1996), pp. 1-13.
18. Dorigo M., Gambardella, L. M., Ant Colony System: a Cooperative Learning Approach to the Traveling Salesman Problem, *IEEE Transactions on Evolutionary Computation*, Vol. 1 (1997), 53-66.
19. Sandou G., Font S., Tebbani S., Huret A., Mondon C., Optimisation par colonies de fourmis d'un site de génération d'énergie, *Journal Européen des Systèmes Automatisés*, Numéro spécial Métaheuristiques pour l'optimisation difficile, Vol. 38, n°9/10 (2004), pp. 1097-1119.
20. Stützle T., Hoos, H. H., MAX-MIN Ant System, *Future Generation Computer Systems*, Vol. 16 (2000), pp. 889-914.
21. Socha K., Dorigo M., Ant colony optimization for continuous domains, Accepted to special issue of *EJOR* on adapting metaheuristics to continuous optimization (2006).
22. Sandou G., Modélisation, optimisation et commande de parcs de production multi énergies complexes, PhD. Thesis, University Paris XI, (2005).

A Shuffled Complex Evolution of Particle Swarm Optimization Algorithm

Jiang Yan¹, Hu Tiesong¹, Huang Chongchao², Wu Xianing¹, and Gui Faling¹

¹ State Key Laboratory of Water Resources and Hydropower Engineering Science, Wuhan University, Wuhan 430072, China

lirenjy@sohu.com, tshu@whu.edu.cn, wxn800228@sohu.com, falinggui@sina.com

² School of Mathematics and Statistics, Wuhan University, Wuhan, 430072, China
huangchongchao@hotmail.com

Abstract. A shuffled complex evolution of particle swarm optimization algorithm called SCE-PSO is introduced in this paper. In the SCE-PSO, a population of points is sampled randomly in the feasible space. Then the population is partitioned into several complexes, which is made to evolve based on PSO. At periodic stages in the evolution, the entire population is shuffled and points are reassigned to complexes to ensure information sharing. Both theoretical and numerical studies of the SCE-PCO are presented. Five optimization problems with commonly used functions are utilized for evaluating the performance of the proposed algorithm, and the performance of the proposed algorithm is compared to PSO to demonstrate its efficiency.

1 Introduction

Particle Swarm Optimization (PSO) is a stochastic, population-based algorithm for solving optimization problems, which was introduced in 1995 by Eberhart and Kennedy [1] [2], who was inspired by the social behavior of animals such as fish schooling and bird flocking. PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. At each step, each particle keeps track of its coordinates in hyperspace which are associated with the best solution it has achieved so far. This value is called *pbest*. And it keeps also track of the overall best value, and its location, obtained thus far by any particle in the population; the best value is a global best and is called *gbest*.

As general swarm intelligence method, PSO also has premature convergence, especially in complex multi-peak-search problems. Many researchers are devoted into this field to handle its premature convergence. Angeline [3] incorporates PSO with an explicit selection mechanism similar to that used in more traditional evolutionary computations to improve PSO. Løvbjerg [4] proposes a hybrid PSO combining the traditional velocity and position update rules with the ideas of breeding and subpopulations, which has the potential to achieve faster convergence and the potential to find a better solution. Parsopoulos [5] [6] introduces function “stretching” to PSO for the alleviation of the local minima problem. In [7], Parsopoulos uses the

technique of initializing the particle swarm optimizer using the nonlinear simplex method to explore the search space more efficiently and detect better solutions. Higashi [8] presents particle swarm optimization with Gaussian mutation. This method has been proved that can succeed in acquiring the better results than those by PSO alone. Inspired by GA, Shi [9] presents a hybrid evolutionary algorithm based on PSO and GA methods through crossing over PSO and GA, which possess better ability to find the global optimum than that of the standard PSO algorithm. Wang [10] integrates PSO and simulated annealing to improve the performance of PSO.

Inspired by [11], we propose a shuffled complex evolution of particle swarm optimization algorithm (SCE-PSO) in this paper. The paper is organized as follows: Section 2 presents a review of PSO. Description of the proposed algorithm the SCE-PSO is given in section 3. A convergence study of the SCE-PCO is given in section 4. Numerical examples used to illustrate the efficiency of the proposed algorithm are given in section 5. Finally, section 6 concludes this paper.

2 Overview of PSO

PSO is a population based optimization tool, where the system is initialized with a population of random particles and the algorithm searches for optima by updating generations. At each step, each particle keeps track of its coordinates in hyperspace which are associated with the best solution it has achieved so far. And it keeps also track of the overall best value, and its location, obtained thus far by any particle in the population. Suppose that the search space is n -dimensional, and then the particle i of the swarm can be represented by an n -dimensional vector $X_i = (x_{i1}, x_{i2}, \dots, x_{in})$. The velocity of this particle can be represented by another n -dimensional vector $V_i = (v_{i1}, v_{i2}, \dots, v_{in})$. The fitness of each particle can be evaluated according to the objective function of optimization problem. The best previously visited position of the particle i is noted as its individual best position $P_i = (p_{i1}, p_{i2}, \dots, p_{in})$. The position of the best individual of the whole swarm is noted as the global best position $G = (g_1, g_2, \dots, g_n)$. At each step, the velocity of particle and its new position will be assigned according to the following two equations:

$$v_{ij}(t+1) = \chi^* (v_{ij}(t) + c_1 * r_1 * (p_{ij}(t) - x_{ij}(t)) + c_2 * r_2 * (g_j(t) - x_{ij}(t))) \quad (1)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \quad (2)$$

Where, r_1 and r_2 are independently uniformly distributed random variables with range $(0,1)$. c_1 and c_2 are positive constant parameters called acceleration coefficients which control the maximum step size. χ is a constriction factor, which is used to limit the velocity. In PSO, Equation (1) is used to calculate the new velocity according to its previous velocity and to the distance of its current position from both its own best historical position and the best position of the entire population or its neighborhood. Generally, the value of each component in V can be clamped to the range

$[-V_{\max}, V_{\max}]$ to control excessive roaming of particles outside the search space. Then the particle flies toward a new position according equation (2). This process is repeated until a user- defined stopping criterion is reached.

3 Description of SCE-PSO Algorithm

In the original PSO, the swarm is guided by the best solution found so far by the whole swarm, which we call the *gbest* model. In recent years, many neighborhood topologies have been investigated for PSO, which uses each particle’s best current performance of its neighbors to replace the best previous one of the whole swarm, which we call the *lbest* model. In this paper, the algorithm SCE-PSO is proposed, which uses the particle’s so far best found function value to define the neighborhood relations. This algorithm is similar to the *lbest* model in the fact that only a part of the swarm is considered for the velocity update of a particle. But in our algorithm the neighborhoods are constantly changing, according to the fitness development for the individuals. The SCE-PSO strategy is presented below and is illustrated in Fig.1.

Step 1: Initializing. Select $p \geq 1, m \geq 1$, where, p is the number of complexes, m is the number of points in each complex. Compute the sample size $s = pm$. Sample s points X_1, \dots, X_s in the feasible space. Compute the function value f_i at each point X_i .

Step 2: Ranking. Sort the points in order of increasing function value. Store them in an array $E = \{X_i, f_i, i = 1, \dots, s\}$.

Step 3: Partitioning. Partition E into p complexes A^1, A^2, \dots, A^p , each containing points m , such that: $A^k = \{X_j^k, f_j^k \mid X_j^k = X_{k+p(j-1)}, f_j^k = f_{k+p(j-1)}, j = 1, \dots, m\}$.

Step 4: Evolving. Evolve each complex A^k using PSO separately.

Step 4.1: Initializing. Select q, T , where, q is the population size of PSO, T is the maximal iterated generation.

Step 4.2: Selecting. Choose q distinct points Y_1^k, \dots, Y_q^k from A^k according to the function values to construct a sub-swarm. Better points in A^k have more probability to be selected. Store them in $F^k = \{Y_i^k, V_i^k, u_i^k, i = 1, \dots, q\}$, where V_i^k is the velocity for particle Y_i^k and u_i^k is the corresponding function value. Find out the best previously visited position of each particle P_i^k and the position of the best individual of the complex G^k .

Step 4.3: Comparing. Compare the function value between each particle Y_i^k and P_i^k . If Y_i^k is better than P_i^k , then $P_i^k = Y_i^k$. Compare the function value between each particle Y_i^k and G^k . If Y_i^k is better than G^k , then $Y_i^k = G^k$.

Step 4.4: Renewing. According to the formulation (1) and (2), renew the position and velocity of each particle.

Step 4.5: Iterating. Iterate by repeating **Step 4.3** and **Step 4.4** T times, where T is a user-specified parameter which determines how fast each complex should evolve.

Step 5: Complexes shuffling. Replace A^1, \dots, A^p into E . Sort E in order of increasing function value.

Step 6: Check convergence. If the convergence criteria are satisfied, stop. Otherwise, return to **Step 4**.

The SCE-PSO combines the strengths of the particle swarm optimization, competitive evolution and the concept of complex shuffling. It greatly enhances survivability by a sharing of the information gained independently by each complex. In the SCE-PSO, each member of a complex is a potential parent with the ability to participate in a process of evolution. A sub-swarm selected from the complex is like a pair of parents. To ensure that the evolution process is competitive, we require that the probability that better parents contribute to the generation of offspring is higher than that of worse parents. Finally, each new offspring replaces the worst point of the current sub-swarm, rather than the worst point of the entire population. This ensures that process before being replaced or discarded. Thus, none of the information contained in the sample is ignored.

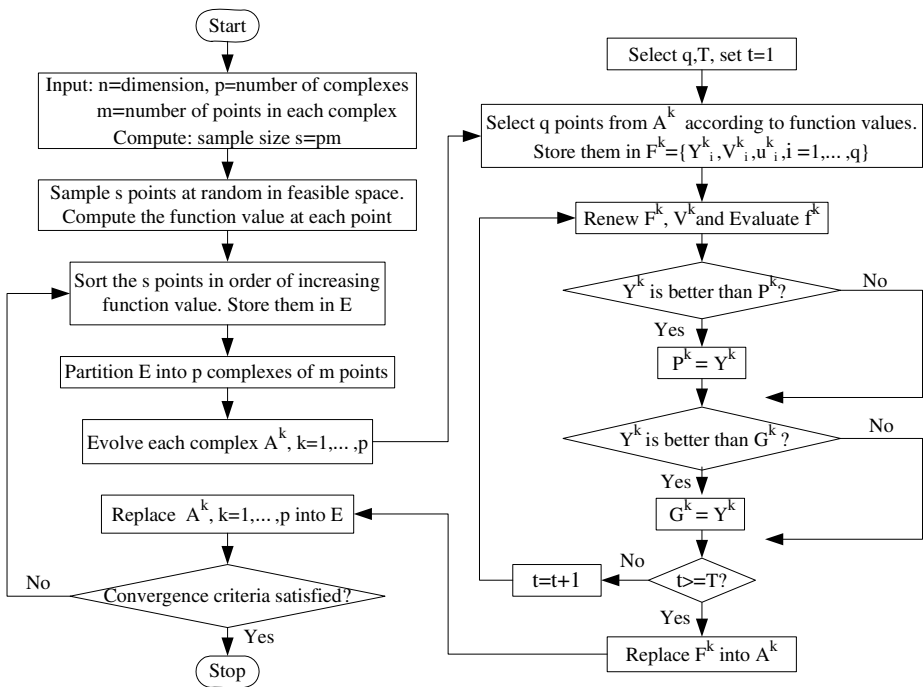


Fig. 1. The flow chart of the proposed SCE-PSO algorithm

4 Convergence Analysis

It is important to note if the trajectory of the particle convergences. Therefore, a convergence study of the SCE-PCO is given in this section. In equation (1) and (2),

the implicit form of the velocity and position equations are used for multi-particles working in a multi-dimensional search space. For ease of notation, the analysis below will be restricted to a single dimension so that the subscript j is dropped. This can be done without loss of generality to dropping the subscript i , since there is no interaction between the different dimensions in the PSO.

Theorem 1. Assume that at time t , the position of a particle is $x(t)$ in the SCE-PSO, and then we will have:

$$\lim_{t \rightarrow \infty} x(t) = q \tag{3}$$

Where q is an arbitrary position in search space.

Proof: According to the literature [12], we can get the following result:

$$\lim_{t \rightarrow \infty} x(t) = \lim_{t \rightarrow \infty} ((1 - \alpha)p(t) + \alpha g(t)) \tag{4}$$

Where $\alpha = c_1/c_1 + c_2$, c_1 and c_2 are acceleration coefficients. $p(t)$ is noted as its individual best position of this particle in each complex $A^k, k = 1, 2, \dots, p$ and $g(t)$ is the best position of the complex at time t , where p is the number of the complexes.

If $c_1 = c_2$, then the equation becomes:

$$\lim_{t \rightarrow \infty} x(t) = \lim_{t \rightarrow \infty} \frac{p(t) + g(t)}{2} \tag{5}$$

Based on PSO, if the new position of the particle in each complex is better than its individual best position, the value of the latter will be replaced by the former. In the same way, if its new position is better than the best position of the complex, the former replaces the latter. Suppose that the final best position of the complex is q , then we have:

$$\lim_{t \rightarrow \infty} g(t) = q \tag{6}$$

Due to update equation (2), the individual best position of the particle will gradually move closer to the best position of the complex, that is:

$$\lim_{t \rightarrow \infty} p(t) = q \tag{7}$$

Substituting equation (6) and (7) into equation (5), we have:

$$\lim_{t \rightarrow \infty} x(t) = q \tag{8}$$

5 Computational Experiences

In this section, five nonlinear benchmark functions that are commonly used in literature [13] are performed. The functions, the number of dimensions, the

Table 1. Optimization test functions

| Name | Formula | Dim | Range | Opt. | Goal |
|---------------|--|-----|------------------|------|-----------|
| Sphere | $f_0(x) = \sum_{i=1}^n x_i^2$ | 30 | $[-100,100]^n$ | 0 | 0.01 |
| Rosenbrock | $f_1(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$ | 30 | $[-30,30]^n$ | 0 | 100 |
| Rastrigrin | $f_2(x) = \sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i) + 10)$ | 30 | $[-5.12,5.12]^n$ | 0 | 100 |
| Griewank | $f_3(x) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ | 30 | $[-600,600]^n$ | 0 | 0.1 |
| Schaffer's f6 | $f_6(x) = 0.5 + \frac{(\sin \sqrt{x_1^2 + x_2^2})^2 - 0.5}{[1.0 + 0.001(x_1^2 + x_2^2)]^2}$ | 2 | $[-100,100]^n$ | 0 | 10^{-5} |

admissible range of the variable, the optimum and the goal values are summarized in Table 1. All functions are designed to have minima at the region.

To evaluate the performance of the proposed SCE-PSO, the basic PSO are used for comparisons. For these two methods, the parameter set ($\chi = 0.729, \chi * c_1 = \chi * c_2 = 1.494$) was recommended by Clerc [14]. In our case for the SCE-PSO, the following values are used as default: $p = 4, m = 2q, T = 3n$, where p is the number of complexes, m is the number of points in each complex, q is selected as the population size of each complex to execute PSO, T is the number of evolution steps taken by each PSO in the SCE-PSO, and n is the dimension size. Each optimization experiment randomly initializes x and v in the range $[x_{\min}, x_{\max}]$ indicated in Table 1. Population sizes of $N = 40, 80, 120$ particles were tested.

In the experiments, the goal value for each function was recorded and the average, maximum and minimum function value was calculated for each algorithm. If the goal was not reached within the maximum number of 10000 iterations, the run was considered unsuccessful. For evaluating the significance, the Wilcoxon Rank Sum Test was used to compare the results for PSO and the SCE-PSO. The results of the 20 test runs for the two algorithms form two independent samples. For two algorithms (A and B), the distribution of their results, F_A and F_B are compared using the null hypothesis $H_0 : F_A = F_B$ and the alternative $H_1 : F_A < F_B$. The tests are made at a significance level of $\alpha = 0.01$. In the results section the significance comparison between the two algorithms is displayed using a 2×2 matrix $A = (a_{ij})_{i,j \in \{1,2\}}$, in which an entry “+” at position a_{ij} denotes that algorithm i is significantly better. For example, the leftmost “+” in the second row of Table 2 indicates that the SCE-PSO is significantly better than PSO. An entry “-” at position a_{ij} denotes that algorithm i is not significantly better than algorithm j . The entries a_{ii} on the main diagonal are left empty. We call the corresponding matrix a significance matrix (s-matrix).

Table 2. The Maximum, Minimum and Average function values of the SCE-PSO and PSO

| Size | Alog. | Max | Min | Avg | Success rate (%) | S-Matrix | |
|---------------|---------|----------|----------|----------|------------------|----------|---|
| Sphere | | | | | | 1 | 2 |
| 40 | PSO | 9.94 | 2.03E-04 | 1.82 | 10 | | - |
| | SCE-PSO | 8.87E-06 | 1.95E-08 | 2.88E-06 | 100 | + | |
| 80 | PSO | 6.42E-01 | 8.90E-05 | 9.34E-02 | 50 | | - |
| | SCE-PSO | 3.67E-12 | 2.50E-13 | 1.03E-12 | 100 | + | |
| 120 | PSO | 5.57E-05 | 1.67E-08 | 8.83E-06 | 100 | | - |
| | SCE-PSO | 1.29E-13 | 4.25E-14 | 7.02E-14 | 100 | + | |
| Rosenbrock | | | | | | 1 | 2 |
| 40 | PSO | 488.22 | 8.19 | 161.54 | 55 | | - |
| | SCE-PSO | 2.89 | 4.33E-06 | 3.08E-01 | 100 | + | |
| 80 | PSO | 5.36E+01 | 1.05E-01 | 1.00E+01 | 100 | | - |
| | SCE-PSO | 5.42E-02 | 8.57E-07 | 1.04E-02 | 100 | + | |
| 120 | PSO | 2.76E-01 | 1.88E-09 | 2.80E-02 | 100 | | - |
| | SCE-PSO | 1.54E-05 | 1.58E-10 | 1.57E-06 | 100 | + | |
| Rastrigrin | | | | | | 1 | 2 |
| 40 | PSO | 150.07 | 54.94 | 105.63 | 40 | | - |
| | SCE-PSO | 27.86 | 1.24 | 20.94 | 100 | + | |
| 80 | PSO | 120.49 | 4.09 | 60.50 | 85 | | - |
| | SCE-PSO | 16.91 | 1.17E-08 | 5.04 | 100 | + | |
| 120 | PSO | 93.74 | 1.53E-05 | 28.74 | 100 | | - |
| | SCE-PSO | 8.95 | 1.28E-13 | 2.64 | 100 | + | |
| Griewank | | | | | | 1 | 2 |
| 40 | PSO | 1.97 | 8.53E-03 | 1.22 | 5 | | - |
| | SCE-PSO | 3.89E-02 | 7.34E-04 | 1.33E-02 | 100 | + | |
| 80 | PSO | 1.43E-01 | 1.16E-04 | 3.83E-02 | 95 | | - |
| | SCE-PSO | 7.82E-05 | 2.23E-12 | 1.91E-05 | 100 | + | |
| 120 | PSO | 6.42E-02 | 2.41E-06 | 7.82E-03 | 100 | | - |
| | SCE-PSO | 5.17E-12 | 2.55E-15 | 3.89E-13 | 100 | + | |
| Schaffer's f6 | | | | | | 1 | 2 |
| 40 | PSO | 0 | 0 | 0 | 100 | | - |
| | SCE-PSO | 0 | 0 | 0 | 100 | - | |
| 80 | PSO | 0 | 0 | 0 | 100 | | - |
| | SCE-PSO | 0 | 0 | 0 | 100 | - | |
| 120 | PSO | 0 | 0 | 0 | 100 | | - |
| | SCE-PSO | 0 | 0 | 0 | 100 | - | |

The performance of the SCE-PSO and PSO are shown in Tables 2 and 3. In most cases, increasing the number of particles decreases both the maximum, minimum, average values and the standard deviation of function values. Whatever the population size is, the function values and standard deviation of function values obtained by the SCE-PSO is smaller than or equal to that of PSO and the success rate of the SCE-PSO (fraction of the number of runs it reached the goal) is superior to that of PSO. These mean that the SCE-PSO has better global convergence and stability than PSO. Table 3 lists mean values of CPU time of the two algorithms. It can be seen that the proposed

Table 3. The deviation of function values and CPU time of the SCE-PSO and PSO

| Function | Pop. size | Deviation of function values | | CPU(s) | |
|---------------|-----------|------------------------------|---------|--------|---------|
| | | PSO | SCE-PSO | PSO | SCE-PSO |
| Sphere | 40 | 2.583 | 0.000 | 20 | 4 |
| | 80 | 0.178 | 0.000 | 66 | 10 |
| | 120 | 0.000 | 0.000 | 137 | 17 |
| Rosenbrock | 40 | 164.709 | 0.71 | 28 | 10 |
| | 80 | 12.688 | 0.019 | 89 | 21 |
| | 120 | 0.063 | 0.000 | 194 | 34 |
| Rastrigrin | 40 | 28.485 | 7.919 | 24 | 6 |
| | 80 | 28.205 | 5.045 | 83 | 13 |
| | 120 | 27.561 | 2.647 | 158 | 23 |
| Griewank | 40 | 0.427 | 0.001 | 27 | 8 |
| | 80 | 0.039 | 0.000 | 86 | 16 |
| | 120 | 0.016 | 0.000 | 160 | 27 |
| Schaffer's f6 | 40 | 0.000 | 0.000 | 4 | 1 |
| | 80 | 0.000 | 0.000 | 9 | 2 |
| | 120 | 0.000 | 0.000 | 14 | 3 |

algorithm the SCE-PSO can find global or near global optimal solutions for all test problems in a short time. Therefore, the proposed algorithm is efficient and effective.

6 Conclusions

This paper presents a shuffled complex evolution of particle swarm optimization algorithm called SCE-PSO designed to improve the performance of PSO. In the SCE-PSO, a population of points sampled randomly from the feasible space. Then the population is partitioned into several complexes, which is made to evolve based on particle swarm optimization (PSO). At periodic stages in the evolution, the entire population is shuffled and points are reassigned to complexes to ensure information sharing.

This new method can enhance survivability by a sharing of the information. Five benchmark functions are performed in the simulation part using different algorithms. The performance comparisons indicate that the SCE-PSO is superior to PSO.

Acknowledgments. This research is supported by National Natural Science Foundation of China (50479039).

References

1. Kennedy, J., Eberhart, R. C.: Particle swarm optimization. Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ. (1995) 1942-1948
2. Eberhart, R. C., Kennedy, J.: A new optimizer using particle swarm theory. Proceedings of the Sixth International Symposium on Micromachine and Human Science, Nagoya, Japan. (1995) 39-43

3. Angeline, P. J.: Using selection to improve particle swarm optimization. Proceedings of the IEEE Congress on Evolutionary Computation (CEC 1998), Anchorage, Alaska, USA. (1998) 84-89
4. Løvbjerg, M., Rasmussen, T. K., Krink, T.: Hybrid particle swarm optimiser with breeding and subpopulations. Proceedings of the Genetic and Evolutionary Computation Conference. (2001)
5. Parsopoulos, K. E., Plagianakos, V. P., Magoulas, G. D., Vrahatis, M. N.: Improving particle swarm optimizer by function "stretching". Advances in Convex Analysis and Global Optimization. (2001) 445-457
6. Parsopoulos, K. E., Plagianakos, V. P., Magoulas, G. D., Vrahatis, M. N.: Stretching technique for obtaining global minimizers through particle swarm optimization. Proceedings of the Workshop on Particle Swarm Optimization, Indianapolis, IN. (2001)
7. Parsopoulos, K. E., Vrahatis, M. N.: Initializing the particle swarm optimizer using the nonlinear simplex Method. Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, WSEAS Press. (2002) 216-221.
8. Higashi, N., Iba, H.: Particle swarm optimization with gaussian mutation. Proceedings of the IEEE Swarm Intelligence Symposium 2003, Indianapolis, Indiana, USA.8 (2003) 72-79
9. Shi, X., Lu, Y., Zhou, C., Lee, H., Lin, W., Liang, Y.: Hybrid evolutionary algorithms based on PSO and GA. Proceedings of IEEE Congress on Evolutionary Computation 2003, Canbella, Australia. 9 (2003) 2393-2399
10. Wang, X. H., Li, J. J.: Hybrid particle swarm optimization with simulated annealing. Proceedings of the Tllid International Conference on Machine Learning and Cybernetics, Shanghai. (2004) 2402-2405
11. Duan, Q. Y., Sorooshian, S., Gupta, V. K.: Effective and efficient global optimization for conceptual rainfall-runoff models. Water Resources Research. 28 (1992) 1015 -1031
12. Van den Bergh F.: An analysis of particle swarm optimizers. Pretoria, Department of Computer Science, University of Pretoria, South Africa. (2002)
13. Trelea, I. C.: The particle swarm optimization algorithm: convergence analysis and parameter selection. Information Processing Letters. 85 (2003) 317-325
14. Clerc, M. The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. Proceedings of the IEEE Congress on Evolutionary Computation. (1999) 1951-1957

Wasp Swarm Algorithm for Dynamic MAX-SAT Problems

Pedro C. Pinto^{1,2}, Thomas A. Runkler², and João M.C. Sousa²

¹ Siemens AG, Corporate Technology, Information and Communications, CT IC 4
81730 Munich - Germany

{pinto.pedro.ext, Thomas.Runkler}@siemens.com

² Technical University of Lisbon, Instituto Superior Técnico
Av. Rovisco Pais, 1049-001 Lisbon - Portugal
jmsousa@ist.utl.pt

Abstract. This paper proposes a wasp swarm optimization algorithm, which is applied to the dynamic variant of the maximum satisfiability problem, or MAX-SAT. Here, we describe the changes implemented to optimize the dynamic problem and analyze the parameters of the new algorithm. Wasp swarm optimization accomplishes very well the task of adapting to systematic changes of dynamic MAX-SAT instances derived from static problems, and significantly outperforms the local search algorithm used as benchmark.

1 Introduction

In this article we discuss an application of wasp swarm optimization (WSO) [1] to the dynamic variant of the constraint satisfaction problem MAX-SAT. By applying WSO to the dynamic variant of MAX-SAT we are solving a problem with different characteristics, specially due to the adaptation required from the algorithm during the optimization process, to which meta-heuristical approaches are particularly suited. For validating of the results we use a version of local search (LS) [2], which is very simple but fulfils the role intended here of showing that WSO is adequate for this problem.

This paper is divided as follows. First, we do a brief introduction to the propositional satisfiability problem, MAX-SAT and dynamic MAX-SAT. Then we present our models of LS and WSO applied to the dynamic SAT. We end the paper with a general conclusion on how well WSO applied to dynamic SAT, and an outline for future work.

2 The MAX-SAT Problem

The general constraint satisfaction problem (CSP) [3] consists of finding an assignment to a set of variables that satisfies a set of constraints over the domains of these variables. This problem has many real-life applications, for example in resource allocation and scheduling, which makes it very interesting as the procedures developed can be easily applied to these problems. A CSP is formally defined as a triple (X, D, C) where $X = \{x_1, \dots, x_n\}$ is the set of variables, $D = \{D_1, \dots, D_n\}$ is the set of domains, which defines the values a variable can assume and $C = \{C_1, \dots, C_m\}$ is the set of constraints among the variables. The defined triple is called an *instance* and represented

by Φ . A solution of the CSP is a complete assignment of the variables that satisfy all the constraints. The MAX-CSP is a variant of the CSP, where the problem may be unsatisfiable, and the goal is then to find the solution that satisfies the maximum number of constraints possible.

The maximal satisfiability problem (MAX-SAT) is a typical \mathcal{NP} -Hard CSP [4] in which the domain of each variable is true or false, or equivalently $X \in \{0, 1\}^n$. Given a set of clauses, each of which is the logical disjunction of $K \geq 2$ variables, the goal is to find an assignment that satisfies the largest number of clauses or, in the weighted version, the assignment that maximizes the sum of weights of the satisfied clauses.

The 3-satisfiability problem, or 3-SAT, is a special case of the k-satisfiability (k-SAT) problem, where each clause contains at most $K = 3$ literals.

3 The Dynamic MAX-SAT Problem

The dynamic SAT problem considered here was introduced in [5] by Holger H. Hoos and Kevin O'Neill and it is a generalization of the satisfiability problem in propositional logic which allows changes of a problem over time. There is more than one way of creating those changes. The possibility that we explored here is to start with a fixed set of clauses and allow certain propositional variables to be set arbitrarily to true or false at different points in time. Each subproblem created in this way is called a stage with an instance $\Phi(i)$ defined in the same way as for the static SAT problem, so they can be directly compared.

The problem can be defined as an instance defined by a set of clauses with a fixed number of variables being set forcibly to true or false in every stage, which means that for the MAX-SAT problem the variable set X is defined in instance $\Phi(i)$ as $x \in \{0, 1, free\}$, where *free* means that the variable is unassigned.

As an example, we can formulate a simple problem. Let us have a static MAX-SAT problem composed by an instance Φ with 3 clauses and 5 variables

$$\Phi = (x_1 \vee \bar{x}_2 \vee x_4) \wedge (x_3 \vee x_4 \vee \bar{x}_5) \wedge (x_1 \vee x_3 \vee \bar{x}_4) \quad (1)$$

Ψ is the function that determines which ones of the variables of X are fixed for each instance and their values, from the set of existing variables: $X_1 = \Psi(X_1)$. In this example, it fixes 2 variables randomly at each stage

$$Stage\ 1 : X = [0, free, 1, free, free] \quad (2)$$

In order to assure the variability of the problem, at least one of the variables fixed in stage i cannot have been fixed in stage $i - 1$.

$$Stage\ 2 : X = [0, free, free, free, 1] \quad (3)$$

$$Stage\ 3 : X = [free, 1, free, free, 1]$$

$$Stage\ 4 : ..$$

Stage 1 can then be formulated as a modified set of constraints and variables. The new constraints being $x_1 = 0$ and $x_3 = 1$, and the variables being x_2, x_4 and x_5 . The assignment $x_2 = 1, x_4 = 1$ and $x_5 = 0$ solves the problem, since

$$(0 \vee 0 \vee 1) \wedge (1 \vee 1 \vee 1) \wedge (0 \vee 1 \vee 0) = \text{true}. \quad (4)$$

In this case, the cost of the solution is 0 (no clauses are left unsatisfied) and the solution is not unique, which does not concern us since the problem here consists only in finding a best solution.

4 Local Search Applied to the Dynamic MAX-SAT Problem

The local search (LS) [2] used in this paper as a benchmark for the results obtained with WSO is shown in Algorithm 1 and starts with an initial solution for each stage, created from the heuristics, which is improved in each iteration by flipping a variable (i.e. changing its value) that makes the cost of the solution lower (or the utility higher). In the unweighted variant of the MAX-SAT problem we define *cost* as the total number of unsatisfied clauses whose variables are all assigned (i.e., clauses that cannot be satisfied). In the weighted variant, we define *utility* as the sum of every satisfied clause multiplied by its weight. When all clause weights have the value one, the utility is identical to the cost. If this variable is not found for a given number of evaluations, then the solution does not improve any further.

We set the maximum number of evaluations equal to the number of variables.

Algorithm 1. Local search applied to dynamic MAX-SAT

Creation of the initial solution

Solution: $S = s(\eta)$

While the stopping criteria are not satisfied

 Create new solution

$S_{new} = \text{LocalSearch}(\text{solution})$

End optimization after N iterations

The heuristic information η of the SAT problem considered here is given by

$$\eta = \frac{1}{1 + f(s_k \cup \langle j, e \rangle) - f(s_k)} \quad (5)$$

where $f(s_k \cup \langle j, e \rangle)$ is the cost of the partial solution s_k of the problem after assigning the value e to variable j . The probability of assigning the value e to variable j is determined by $\eta_{e=0}$ and $\eta_{e=1}$ as in:

$$P_1 = \frac{\eta_{e=1}^\alpha}{\eta_{e=0}^\alpha + \eta_{e=1}^\alpha} \quad (6)$$

where α is a system parameter.

LS evaluates the cost of $(s_k \cup \langle j, e \rangle)$ for the random variable j and its random assignment e . Each iteration of the algorithm returns a new solution that is necessarily either better or the same (in case the local search was exhausted) as the one before. The cost and utility are defined as before.

5 Wasp Swarm Optimization Applied to the Dynamic MAX-SAT Problem

In this section, we introduce the application of the wasp swarm optimization (WSO) to the dynamic constrained satisfiability problem.

Theraulaz introduced in [6] a model for the organizational characteristic of a wasp colony. In addition to the tasks of foraging and brooding, also shared with ant colonies, wasp colonies organize themselves in a hierarchy through interaction between the individuals. This hierarchy is an emergent social order resulting in a succession of wasps from the most dominant to the least dominant and is the inspiration of the WSO algorithm. The model of Theraulaz describes the nature of interactions between individual wasps and their local environment with respect to task allocation.

Cicirello and Smith [1] have first implemented WSO in order to optimize the job assignment problem in factories. The optimization process is sketched in Algorithm 2. In WSO, artificial wasps compete against each other in a tournament in order to win and thus achieve a desired goal. In applying this concept to the SAT problem, the desired goal is defined as the wish to be used as the template in the creation of a new wasp. Each wasp has a certain force F_i that determines its chance of winning the tournament. The force of solution i is defined by cost_i , or Utility $_i$ for the weighted problems.

$$F_{i_{unweighted}} = c_1 \times \frac{1}{\text{cost}_i} + c_2 \times \text{div}_i \quad (7)$$

$$F_{i_{weighted}} = c_1 \times \text{Utility}_i + c_2 \times \text{div}_i \quad (8)$$

where c_1 and c_2 are parameters of the system and div_i is the diversity of solution i . Diversity is defined as the lowest Hamming distance (number of bits which differ between two binary strings) between solution i and all the solutions computed previously, normalized by dividing the distance by the number of variables of the system. The concept of diversity was used in [7] by R. Battiti and M. Protasi to adapt a reactive local search algorithm with good results.

The initial population of solutions is created through the heuristics in the same manner as described before for the local search.

The new solution is created through a stochastic process where the value e of each variable j of the best solution S so far and a heuristically assigned value for the same variable are balanced to obtain a new value. We biased the heuristics η towards the value of the best solution, by a multiplying factor W_s :

$$\eta(e = 0, 1) = \begin{cases} \eta_{0,1} * W_s & \text{if } S_j = 0, 1 \\ \eta_{0,1} & \text{otherwise} \end{cases} \quad (9)$$

With this approach, each solution has a large probability of being created from the immediately previous one, but that is not certain, creating a tree of possible intermediary solutions which are not explored, but remain available to be explored at a later moment with a certain probability, that decreases with time, of being picked. The value

Algorithm 2. MAX-SAT optimization with WSO

Creation of the initial **wasp swarm**
 Solution: $S_i = s(\eta)$
 Force: $F_i = f(S_i)$
 Wasp: $W_i = [S_i, F_i]$
 While the stopping criteria are not satisfied
 Create **new wasp**
 Tournament: $P(W_i) = f(F_i / \sum F)$
 Create **new solution**:
 $S_{new} = s(\eta, P(W_i))$
 Compute **Force** and form **new wasp**
 Update wasp swarm
 Update **tabu list** T
 Update **best solution**
 End optimization

e assigned to variable j is then determined in a similar way as for LS, being 1 with probability given by:

$$P_1 = 1 / \left(1 + \frac{\eta_{e=0}^\alpha}{[\eta_{e=1} * W_s]^\alpha} \right) \quad (10)$$

and 0 otherwise.

The parameters were set to $\alpha = 6$, $P_0 = 1$ and $W_s = 2$.

6 Results for the Dynamic MAX-SAT Problem

We applied our adaptation of wasp swarm optimization to the unweighted version of the dynamic MAX-SAT problem, using as basis for comparison one weighted instance with 100 variables and 850 clauses (jnh1) and 900 clauses (jnh301). Then we applied the algorithms to the unweighted version, using one instance with 250 variables and 1065 clauses (uuf250-01 and uuf250-02). The weighted and unweighted instances can be retrieved respectively from <http://www.research.att.com/~mgcr/data/maxsat.tar.gz> and <http://www.intellektik.informatik.tu-darmstadt.de/SATLIB>. These four instances were made dynamic by building 100 stages with 50 (for the unweighted problems) and 100 (for the weighted problems) iterations per stage with the procedure described in section 3. The number of fixed variables per stage was set to 6. Figure 1 shows the results for LS and WSO applied to the weighted problems, while Figure 2 shows the results for the unweighted problems. For the *jnh* instances we report the average fitness from the optimal solution and the average number of iterations until the best solution is found. For the *uuf* instances we report the average in number of unsatisfied clauses instead of the average fitness. In the weighted case, fitness is defined as

$$fitness = \frac{Utility}{\sum (Weight_{Clause_i})} \quad (11)$$

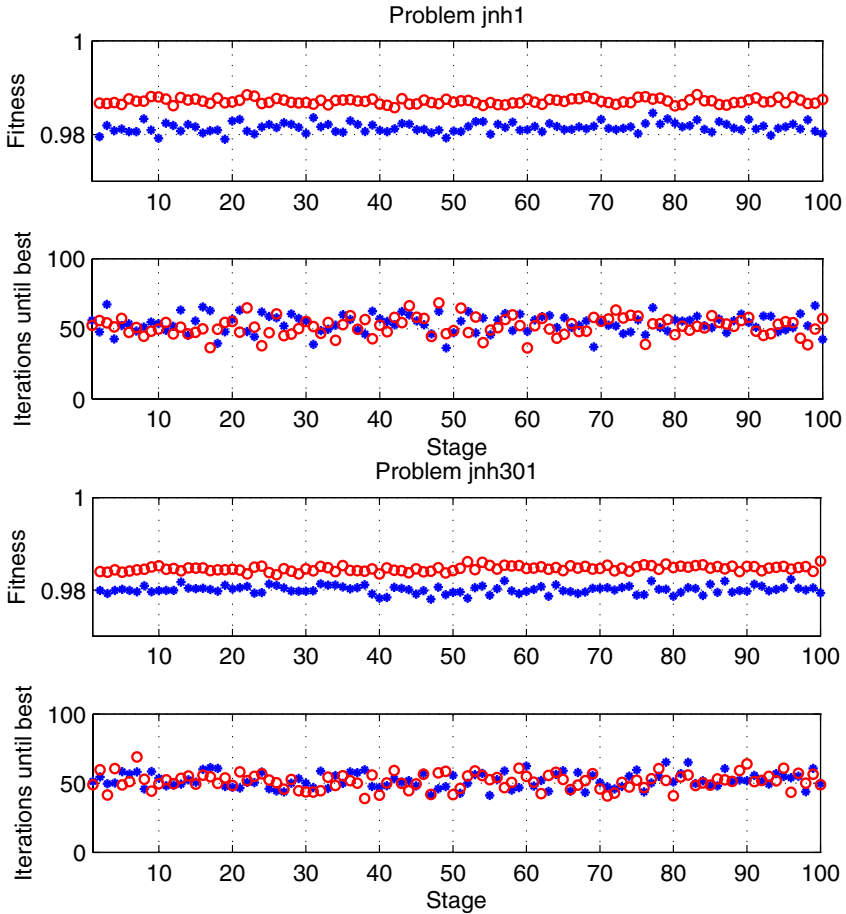


Fig. 1. LS(*) and WSO (o) algorithms applied to the weighted dynamic MAX-SAT problems jnh1 and jnh301, 100 stages each

From the results we conclude that for both the weighted and the unweighted problems WSO easily outperforms LS. The standard deviation for the fitness is more or less identical for both algorithms and the difference is too small to influence the comparison of the algorithms. For the unweighted problems, which are of bigger size, it is clear that besides getting to a better solution than LS, WSO also "needs" less iterations to achieve it (i.e computational cost). It is also noticeable that WSO does stop improving through the stages, as opposite to LS. There is a slight increase in quality of the solution find for each stage as the stages progress.

The chosen number of iterations per stage have obviously influenced these results. If each stage is allowed to have a sufficient number of iterations, then LS will always find results similar to WSO, although with a much higher computational cost. If each stage only has a small number of iterations then the effect seen in the figures still occurs, but the differentiation between LS and WSO only happens after a very large number of

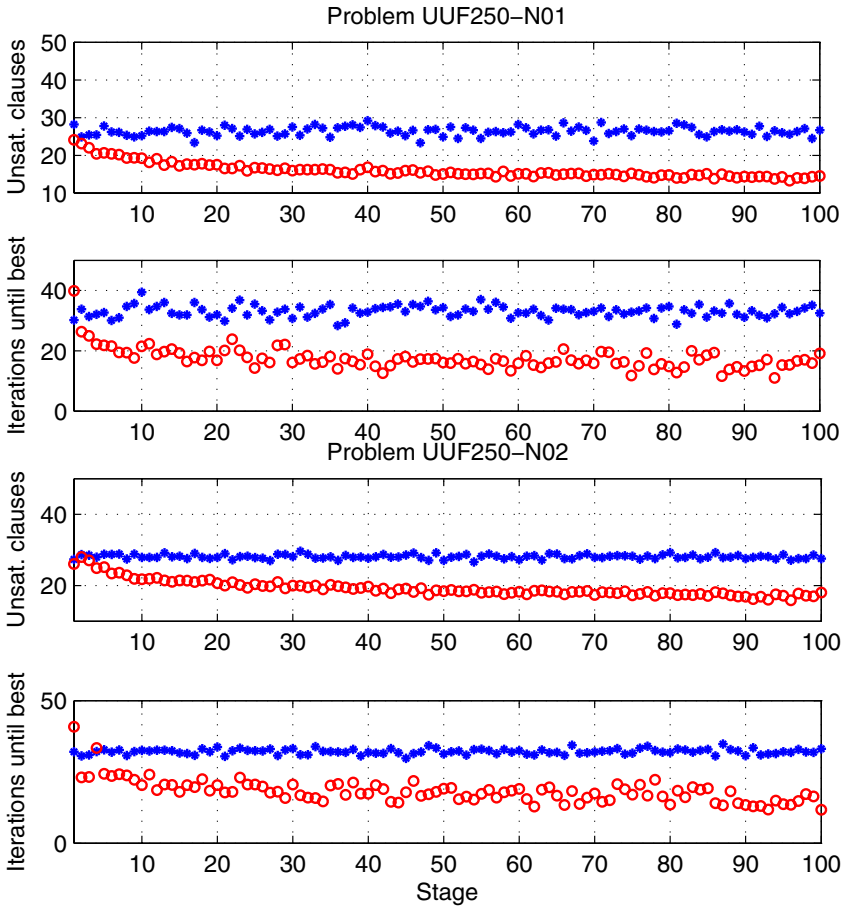


Fig. 2. LS(*) and WSO (o) algorithms applied to the unweighted dynamic MAX-SAT problems uuf250-01 and uuf250-02, 100 stages each

stages, since WSO needs time to get to a visibly better solution. Afterwards, the quality of the solution found will only get better through the stages in comparison with LS, which is unable to find better solutions at a much earlier stage.

7 Conclusions

In conclusion, WSO works very well for the type of dynamic SAT problems analyzed here and the quality of the solutions it finds is better than the quality of the solutions found with the LS algorithm. That is true for both the weighted and unweighted variants of MAX-SAT, with no extra parameter tuning necessary. The quality of the best solution in each stage is always high, with even a tendency to increase through the stages. For future work it is our intention to expand these results to other types of problems. Here we limited the analysis to the problems analyzed in [8].

How WSO compares with other algorithms used to solve the same problem is another question. LS can be improved with better heuristics that try to avoid falling into local minima caused by saturation of LS [9]. These intelligent local search strategies, such as WalkSAT [10] and IROTS [11], provide very good results in the SAT problem. The adopted LS provides a reasonable benchmark in this first evaluation of the use of WSO to optimize dynamic SAT problems.

Acknowledgements. This work is supported by the Portuguese foundation for Science and Technology (FCT) under Grant no, SFRH /BD / 17869 / 2004 and by the project POCI/EME/59191/2004, co-sponsored by FEDER, Programa Operacional Ciência e Inovação 2010, FCT, Portugal.

References

1. Cicirello, V.A., Smith, S.F.: Wasp-like agents for distributed factory coordination. *Autonomous Agents and Multi-agent systems* **8** (2004) 237–266
2. Hoos, H.H., Stützle, T.: *Stochastic Local Search: Foundations and Applications*. Morgan Kaufmann (2004)
3. Tsang, E.: *Foundations of Constraint Satisfaction*. Academic Press (1993)
4. Garey, M.R., Johnson, D.S.J.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*. WH Freeman Publishers (1979)
5. Hoos, H.H., O’Neill, K.: Stochastic local search methods for dynamic SAT - an initial investigation. *AAAI-2000 Workshop "Leveraging Probability and Uncertainty in Computation"*, Austin, Texas (2000) 22–26
6. G. Theraulaz, S. Goss, J.G., Deneubourg, J.L.: Task differentiation in polistes wasps colonies: A model for self-organizing groups of robots. In: *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, (MIT Press, 1991) 346–355
7. Battiti, R., Protasi, M.: Reactive search, a history-based heuristic for MAX-SAT. *ACM Journal of Experimental Algorithmics* **2** (1997)
8. Roli, A., Blum, C., Dorigo, M.: ACO for maximal constraint satisfaction problems. *MIC’2001 - Metaheuristics International Conference*, Porto, Portugal (2001) 187–192
9. Hoos, H.H., Stützle, T.: Local search algorithms for SAT: an empirical evaluation. *Journal of Automated Reasoning*, special Issue "SAT 2000" (1999) 421–481
10. Selman, B., Kautz, H., Cohen, B.: Local search strategies for satisfiability testing. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **26** (1996) 521–532
11. Smyth, K., Hoos, H.H., Stützle, T.: Iterated robust tabu search for MAX-SAT. *Lecture Notes in Computer Science*, Springer Verlag **44** (2003) 279–303

Particle Swarm Optimization for the Multidimensional Knapsack Problem

Fernanda Hemberger, Heitor S. Lopes*, and Walter Godoy Jr.

Federal University of Technology Paraná (UTFPR),
Av. 7 de setembro, 3165 80230-901, Curitiba (PR), Brazil
fernanda@denes.com.br, hslopes@pesquisador.cnpq.br, godoy@utfpr.edu.br

Abstract. The multidimensional 0/1 knapsack problem is a classical problem of discrete optimization. There are several approaches for solving the different variations of such problem, including mathematical programming and stochastic heuristic methods. This paper presents the application of Particle Swarm Optimization (PSO) for the problem, using selected instances of ORLib. For the instances tested, results were very close or equal to the optimal solution known, even considering that the parameters of PSO were not optimized. The analysis of the results suggests the potential of a simple PSO for this class of combinatorial problems.

1 Introduction

There are problems with important practical applications concerned with the search of the “best” configuration or set of parameters to achieve some objective criteria. Such problems are generally referred to as optimization problems. The knapsack problem is a classical optimization problem and has many practical applications in industry, transportation and logistics. A simple and general description of the problem is as follows. Given a set of objects with corresponding values and costs (weights), select some of them to put in a container (knapsack), without extrapolating its capacity, in such a way that maximize the sum of values. This is the basic definition of the problem. However, there are several variations, such that [10]:

- Single knapsack problem: all objects must be put in a single container;
- Multidimensional knapsack problem: more than one container is available;
- Multiple-choice knapsack problem: objects are clustered into subsets and at most one object can be selected;
- Bounded knapsack problem: there is a limited number of objects available to be selected.

This paper is related to the Multidimensional Knapsack Problem (MKP), possibly, the most widely known version of the problem [2]. This same problem

* This work was partially supported by the Brazilian National Research Council – CNPq, under research grant no. 305720/2004-0 to H.S. Lopes.

is also referred as: Multiconstraint, Multi-Knapsack or 0/1 Multidimensional Knapsack Problem.

Many problem-independent and domain-specific heuristics have been developed for optimization problems and, in particular, to the MKP that is NP-complete. The quest for a computational algorithm that are effective both in terms of processing time and quality of the solutions is the motivation of this work. Therefore, we propose a methodology for solving MKP using a relatively recent evolutionary computation technique, the Particle Swarm Optimization (PSO), proposed by Kennedy and Eberhart [7]. This problem has been explored by using genetic algorithms [2], [6], [8], and there are evidences that PSO can be useful for it [4]. PSO, in fact, has been applied successfully to several problems like pattern recognition, classification, scheduling, mobile robotics, image processing, and others [5].

2 Particle Swarm Optimization

PSO is a heuristic method for optimization, inspired in the behavior of social agents found in nature. This behavior can be observed in bird flocking, bee swarming, and fish schooling, for instance.

The computational model is population-based. Agents, or particles, change their position (state) in the multidimensional search space of the problem, according to their own experience and the influence of the neighboring particles. Each particle has a limited store capability, keeping track only of information about its current position, speed and quality (fitness of the solution regarding the problem), as well as its best position ever visited (“best particle solution” – *pbest*). Amongst the swarm of particles, the one with best quality is referred as “the best global solution” – *gbest*. Alternatively, only the neighborhood of the particle is considered, that is, the (“best local solution” – *lbest*. At each time tick, particles move, influenced by both *pbest* and *gbest*, to a new position in the search space. This is an iterative process, repeated until a stop condition is met, usually a predefined number of iterations. Whenever a better solution than the previous is found, *gbest* is updated. This procedure is similar to the principle of elitism, common in other evolutionary computation paradigms, since throughout iterations the best solution is conserved. However, there is a subtle difference: *gbest* is updated is a reference for all particles in the same iteration (in a genetic algorithm, this would be similar to say that all individuals would mate with the best individuals).

It is interesting that *pbest* would be a point with good fitness and also located quite far from *gbest* in the search space, so as to improve diversity. In PSO, like other population-based heuristics, maintaining diversity throughout iterations is often a challenge, and it is a necessary condition to assure a satisfactory exploration of the search space. When many *pbest* ’s are somewhat close to the *gbest*, there will be a particle crowding and the search stagnates. A mechanism to avoid the consequences of this unavoidable convergence is the explosion of the swarm, sending particles to random positions and keeping *gbest*.

In the classical PSO model, the movement of the i -th particle is defined by (1), where its next position in the search space (X_i^{t+1}) is updated using the current position and a speed term (V_i^t):

$$X_i^{t+1} = X_i^t + V_i^t. \tag{1}$$

In fact, the speed term actually does not have the dimension of velocity. It could be better defined as ΔX_i ; but, for the sake of simplicity, it is called speed, following [5]. The speed term, in turn, is defined according to (2):

$$V_i^t = c_1 \cdot r_1 \cdot \Delta p_{best} + c_2 \cdot r_2 \cdot \Delta g_{best}, \tag{2}$$

where: V_i^t is the current speed of the i -th particle; r_1 and r_2 are random values in the range [0..1]; c_1 and c_2 are the weights of p_{best} and g_{best} , respectively (in percentage); Δp_{best} and $d_{g_{best}}$ are the distance between the current position and p_{best} and the current position and g_{best} , respectively. The speed term, that is, the updating rate of the current position, is directly proportional to the distance between the current position to p_{best} and g_{best} . Therefore, within few iterations the particle will be attracted to either p_{best} or g_{best} .

The speed term controls the amount of global and local exploration of the particle (that is, the balance between exploration and exploitation). A high speed facilitates global exploration, while small speed will encourage local search. A user-defined upper bound (V_{max}) is established to limit the maximum speed of particles. Figure 1 shows graphically the elements that influence the position of a particle in the hypersurface of the search space (in this case, a bi-dimensional space).

According to a psychological interpretation of PSO [5], the swarm of particles is like a population of individuals. Then, the two terms of (2) represent the cognitive and the social components of a particle's behavior. The former leads the particle to repeat its own past successful behaviors, while the latter makes it follows the others'. There are no default values for weights c_1 and c_2 ; sometimes

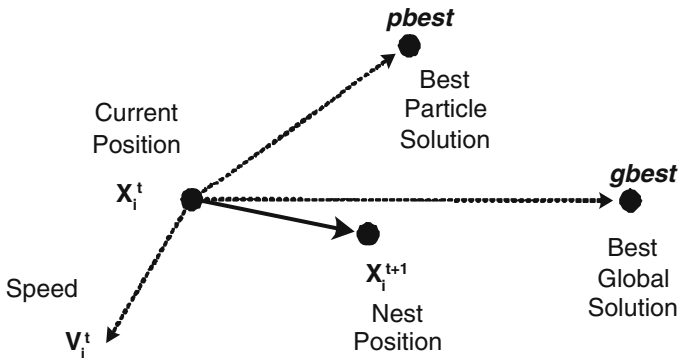


Fig. 1. Illustration of how the position of a particle is updated in PSO

they are set identical and sometimes they are set asymmetrical. It is commonly accepted that those weights are problem-dependent and this seems to be an open subject for further research [3].

3 Methodology

A description of the MKP includes n objects and m knapsacks with specific capacities c_j ($j = 1, \dots, m$). There are binary variables x_i ($i = 1, \dots, n$) that are set to 1 if the i -th object is selected to be put in the knapsacks, and 0, otherwise. Every object has a satisfaction value p_i ($i = 1, \dots, n$) and a specific weight w_{ij} for each knapsack. Therefore, the optimization problem is defined as follows [8, 10]:

$$\begin{aligned} & \text{maximize} && \sum_{i=1}^n p_i \cdot x_i, && (3) \\ & \text{subject to} && \sum_{j=1}^m w_{ij} \cdot x_i \leq c_j. && (4) \end{aligned}$$

The objective of the MKP is to fill the knapsacks with the most valuable objects without extrapolating their capacities. Therefore, a particle should be represented as a possible solution for the MKP. In words, a particle is a binary vector, where each element indicates whether or not an object is selected to be included in the knapsacks. The length of the vector depends on the number of objects available for the selection, that is, it represents the n -dimensional search space. It should be noted that, in this formulation, a given selected object is to be included in all knapsacks.

The evaluation of a possible solution is given by a fitness function. This function is what PSO will optimize. The fitness function was defined before (3), but the constraint (4) is not directly dealt by the algorithm, and so, unfeasible solutions can be found during search. This policy is frequently used in evolutionary algorithms because during evolution towards the best global solution, the algorithm can pass through regions of unfeasible solutions. In PSO, a given particle is dynamically attracted by the social and the cognitive components, and an unfeasible particle now can be changed to a feasible one later. A particle representing an unfeasible solution is allowed to exist in the swarm, but a penalty will be imposed to the fitness of such particle. This penalty is proportional to the total amount of excess in the knapsacks.

4 Computational Experiments

For the computational experiments, we used several instances of MKP found in [1]. These instances were divided into two groups: the first one corresponded to series “sento” [12] and “weing” [15], and the second group, to “weish” [13]. For all experiments reported in this section, the PSO was run for 300 iterations.

The first group of experiments had 10 problems with either 2 or 30 knapsacks and 28, 60 or 105 objects. For these experiments, results are shown in Table 1. In this table, n is the number of objects and m is the number of knapsacks.

Table 1. Comparison of results obtained by PSO and the optimal known solutions

| problem | <i>m</i> | <i>n</i> | optimum | best | abs.diff. |
|---------|----------|----------|---------|---------|-----------|
| sento1 | 30 | 60 | 7772 | 7725 | 0.605% |
| sento2 | 30 | 60 | 8722 | 8716 | 0.069% |
| weing1 | 2 | 28 | 141278 | 138927 | 1.664% |
| weing2 | 2 | 28 | 130883 | 125453 | 4.149% |
| weing3 | 2 | 28 | 95677 | 92297 | 3.533% |
| weing4 | 2 | 28 | 119337 | 116622 | 2.275% |
| weing5 | 2 | 28 | 98796 | 93678 | 5.180% |
| weing6 | 2 | 28 | 130623 | 128093 | 1.937% |
| weing7 | 2 | 105 | 1095445 | 1059560 | 3.276% |
| weing8 | 2 | 105 | 492347 | 492347 | 0.000% |

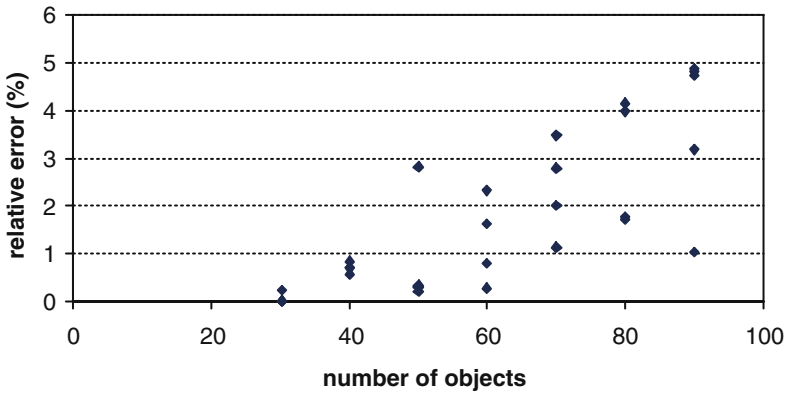


Fig. 2. Performance of PSO regarding the complexity of the problem

Each experiment was repeated 100 times with different random seeds and the best solution found by PSO is shown in the table. Therefore, a total of 1,000 experiments were run in this group.

The second group of experiments was to investigate the behavior of PSO regarding the complexity of the problem. In this particular case, complexity is understood as the number of possible combinations of objects \times knapsacks. The second group of experiments had 30 problems, always with 5 knapsacks, but objects ranged from 30 to 90. Figure 2 shows the performance of PSO regarding the relative difference between the best value found in 100 independent runs and the optimum known value. As mentioned before, for this group of experiments the number of knapsacks is constant (5) and the number of objects increases.

To investigate the convergence of the algorithm, we analyzed the experiments of four different instances with the same number of knapsacks: weish2, weish12, weish23, weish28, having 30, 60, 80, 90 objects, respectively. Figure 3 shows

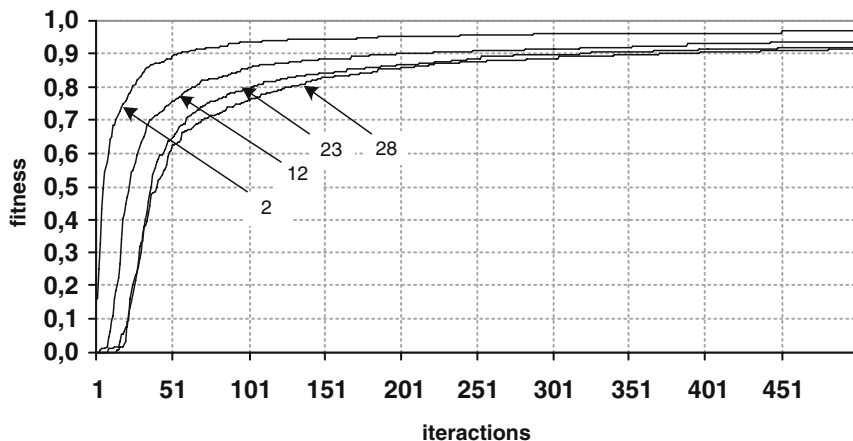


Fig. 3. Evolution of solutions for some instances of the MKP

the evolution of the best solution found by PSO at each iteration for the four instances (numbered as 2, 12, 23 and 28) in 500 iterations. Each point of the curves is the average of the best solution found in 100 independent runs with random seeds.

It should be noted that [2] present a genetic algorithm for the MKP using the instances used in our work, and they always found the optimum. However, to achieve such performance, they needed to process 10^4 chromosomes, thus requesting a large computational effort, not comparable with the PSO implementation.

5 Conclusions and Future Work

In table I, the average performance of PSO for the ten case studies considered was 2.269% of the known optimum. This shows that PSO can perform well for this class of combinatorial problem, even for large instances. That is, PSO seems to be efficient in navigating the hypersurface of the search space and finding good solutions (and, sometimes, the best solution) independently of the initialization and the trajectory of particles.

The second group of experiments aimed at identifying how the performance of PSO degraded as the difficulty of the problem increased. For the particular range of experiments done, we observed that PSO tends to decrease the average performance almost linearly, as shown in figure 2. However, more experiments have to be done so as to confirm this tendency for even harder instances. Also, it can be seen in the graphics that, for the same degree of difficulty, different performances are achieved, depending on the specific nature of each instance.

Figure 3 indicates that, the harder the instance, the longer PSO takes to converge. In the figure, it can be observed a positive derivative in the curves,

suggesting that more iterations, besides 500, are needed to converge. This is specially true for the harder instances.

It is a matter of fact that PSO is sensitive to its control parameters, particularly for hard combinatorial problems with large search space [14]. Recall that no serious attempt was done to optimize the running parameters for the PSO, what could improve the performance achieving better results. Therefore, future work will focus on fine-tuning the PSO parameters for given classes of problems, by using some kind of adaptive strategy [11]. Recent literature have shown that a PSO hybridized with a local search technique certainly can achieve better results than a “pure” PSO, independently of the problem [9]. Therefore, further improvement of the system will be towards the hybridization with some local search technique.

In general, the use of a problem-independent heuristics, such as PSO, gives robustness and efficiency to the exploration of the search space of difficult problems. We believe that this is a promising method for solving several classes of combinatorial problems.

References

1. Beasley, J.E.: ORLib – Operations Research Library. [<http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/mknab2.txt>], (2005).
2. Beasley, J.E., Chu, P.C.: Genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics* **4** (1998) 63–86.
3. Clerc, M.: The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. In: *Proc. IEEE Congress on Evolutionary Computation*, vol. 3, pp. 1951–1957, (1999).
4. Eberhart, R.C., Shi, Y.: Comparison between genetic algorithms and particle swarm optimization. In: *Proc. 7th Annual Conference on Evolutionary Programming*, pp. 611–616, (1998).
5. Eberhart, R.C., Shi, Y.: Particle swarm optimization: developments, applications and resources. In: *Proc. Congress on Evolutionary Computation*, Seoul, Korea, vol. 1, pp. 81–86 (2001).
6. Hoff, A., Løkketangen, A., Mittet, I.: Genetic algorithm for 0/1 multidimensional knapsack problems. In: *Proc. Norsk Informatikkonferanse*, Molde, Norway, (1996).
7. Kennedy, J., Eberhart, R.C.: *Swarm Intelligence*. Morgan Kaufmann, San Francisco, USA (2001).
8. Khuri, S., Bäck, T., Heitkoetter, J.: The zero/one multiple knapsack problem and genetic algorithm. In: *Proc. ACM Symposium on Applied Computing*, Phoenix, USA, pp. 188–193 (1994).
9. Lopes, H.S., Coelho, L.S.: Particle swarm optimization with fast local search for the blind travelling salesman problem. In: *Proc. 5th Hybrid Intelligent Systems Conference*, Rio de Janeiro, Brazil, pp. 245–250 (2005).
10. Martelo, S., Toth P.: *Knapsack Problems – Algorithms and Computer Implementations*. John Wiley & Sons, New York, USA (1990).
11. Maruo, M.H., Lopes, H.S., Delgado, M.R.B.S.: Self-adapting evolutionary parameters: encoding aspects for combinatorial optimization problems. In: Raidl, G.R., Gottlieb, J. (eds.), *Evolutionary Computation for Combinatorial Problems*, LNCS, v. **3448** (2005) pp. 154–165.

12. Senyu, S., Toyoda, Y.: An approach to linear programming with 0-1 variables. *Management Science* **15** (1967) B196–B207.
13. Shih, W.: A branch and bound method for the multiconstraint zero-one knapsack problem. *Journal of the Operational Research Society* **30** (1979) 369–378.
14. Trelea, I.C.: The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters* **85** (2003) 317–325.
15. Weingartner, H.M., Ness, D.N.: Methods for the solution of multi-dimensional 0/1 knapsack problems. *Operations Research* **15** (1967) 83–103.

Particle Swarms for Multimodal Optimization

Ender Özcan and Murat Yılmaz

Yeditepe University, Department of Computer Engineering,
34755 Kadıköy/Istanbul, Turkey
{eozcan,myilmaz}@cse.yeditepe.edu.tr

Abstract. In this paper, five previous Particle Swarm Optimization (PSO) algorithms for multimodal function optimization are reviewed. A new and a successful PSO based algorithm, named as CPSO is proposed. CPSO enhances the exploration and exploitation capabilities of PSO by performing search using a random walk and a hill climbing components. Furthermore, one of the previous PSO approaches is improved incredibly by means of a minor adjustment. All algorithms are compared over a set of well-known benchmark functions.

1 Introduction

Inspiring from the swarms in nature, such as; birds, fish, etc., Kennedy and Eberhart [7] proposed a population based algorithm called Particle Swarm Optimization (PSO). PSO combines *cognition only model* that values solely the self-experience and *social only model* that values solely the experience of neighbors. A particle encodes a candidate solution to a problem at hand. The algorithm uses a set of particles flying over a search space and moving towards a promising area to locate a global optimum. However, there are a set of problems requiring discovery of equal quality candidate solutions, so that, a user could make a choice in between them. In some problems, local optima, or a set of solutions with a predetermined quality levels can also be requested. Multimodal optimization problems represent such class of problems in which the researchers are interested. Different PSO algorithms have been already proposed for solving multimodal problems. These algorithms are mostly based on existing approaches used in the evolutionary algorithms for multimodal optimization.

Most of the real world problems carry multimodal characteristics; hence developing efficient algorithms for multimodal optimization problems is still a research area. Previous approaches can be categorized as *iterative* and *subpopulation* methods [8]. In the iterative methods, the algorithm is applied several times consecutively to locate each optimum. In the subpopulation methods, the population is divided into parts to search optima simultaneously. In this paper, the previous PSO algorithms for multimodal optimization based on subpopulation model are compared to a proposed Particle Swarm Optimizer with craziness and hill climbing, named as CPSO. Additionally, a previous niching PSO approach is modified, yielding an improved performance. All algorithms are described in Sect. 2 and 3. The experimental results are presented in Sect. 4. Finally, the conclusions are provided in Sect. 5.

2 PSO Systems for Multimodal Function Optimization

In a PSO system based on *inertia weight* [6], particles representing candidate solutions start their flight from random locations in a search landscape. At each step, a particle updates its velocity to move to another location based on (1) and (2). The flight is influenced by a *fitness function* that evaluates the quality of each solution.

$$v_{id}(t+1) = w v_{id}(t) + c_1 r_1(t) (y_{id}(t) - x_{id}(t)) + c_2 r_2(t) (y_{gd}(t) - x_{id}(t)), \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1), \quad (2)$$

where $x_{id}(t)$ is the position of the i^{th} particle at time t on dimension d , v is the velocity, w is the inertia weight, c_1 and c_2 are constant values, r_1 and r_2 are uniform random numbers in $[0,1]$, y_i is the i^{th} particle's best position (generating the best fitness) that has been found so far, and y_g is the best position visited by the neighbors. Generally, the neighborhood is chosen as the whole population for global optimization. If c_1 is set to 0 (and $c_2 \neq 0$), the PSO system turns into the social only model and if c_2 is set to 0 (and $c_1 \neq 0$), then the system becomes the cognition only model. In this paper, five multimodal PSO algorithms are discussed that are extended from the generic approach: *Species-based PSO* (SPSO), *Niching PSO* (NichePSO), *nbest PSO* (nbest-PSO), *Unified PSO* (UPSO) and *Parallel Vector-Based PSO* (PVPSO).

Li's [8] *Species-based PSO* gathers the *similar* particles into the sub-swarms called *species*. As a similarity measure Euclidean distance is used. SPSO requires an additional parameter called *species radius*; r_s . The best fit particle in a species is called the *species seed*, and the *boundary* of a species is the circle having the radius of r_s around this seed. The particles in the entire swarm move within their own species at each iteration. Then, they are evaluated and the species are redefined. The multiple optima are maintained in a parallel manner. The convergence rate of the algorithm is enhanced by the communication of particles in the swarm through the PSO algorithm and the reconstruction of the species.

Brits, Engelbrecht and van den Bergh [4] proposed the *nbestPSO*. This method redefines the neighborhood best position to increase the diversity during the information sharing between particles. For each particle i , k nearby particles are determined, and the neighborhood best position y_{gi} is calculated as the center of mass of the best positions visited by these k particles. In (1), y_{gi} replaces y_g . Then the same velocity update equation in (2) is used. Increasing, decreasing and constant k values are analyzed by the researchers. The results show that linearly decreasing k value yields the best performance.

The *Unified PSO*, introduced by Parsopoulos and Vrahatis [13], aims to bring a balance to the global and local variants of PSO. The algorithm requires local and global neighborhoods to be defined. In this algorithm, the velocity update equation ($U_i^{(k+1)}$) is changed and divided into local ($L_i^{(k+1)}$) and global ($G_i^{(k+1)}$) parts. A particle samples two different velocities using two different velocity update PSO equations based on the constriction PSO model, where the constriction factor X controls the velocity's magnitude [6]. The newly introduced *unification factor*; $u \in [0,1]$ determines the effect of the global and local information:

$U_i^{(k+1)} = u G_i^{(k+1)} + (1-u) L_i^{(k+1)}$, where $U_i^{(k+1)}$ is the new location of the i^{th} particle after the k^{th} iteration. Additionally, a normally distributed random parameter is also introduced to be multiplied with either $G_i^{(k+1)}$ or $L_i^{(k+1)}$, yielding two different models that supports mutation.

The *Parallel Vector-Based PSO* (PVPPO), introduced by Schoeman and Engelbrecht [14], uses a set of vector operations to form niches in the search space. PVPPO performs better than their previous approach the *Vector-Based PSO* (VBPSO). In PVPPO, the initial niches are identified as in VBPSO, but all particles are evaluated simultaneously. The velocity update is done using the personal best and the best neighborhood positions. A sub-swarm may absorb the other particles that come close by and/or merge with another one.

2.1 The Niching Particle Swarm Optimizer (NichePSO)

Brits, Engelbrecht and van den Bergh proposed an algorithm as presented in Fig.1 to locate the multiple optima using a particle swarm based algorithm, referred as *NichePSO* [5]. The initial swarm, called as the main swarm is generated by uniformly distributing particles over the search space. The quality of the particles is monitored during the iterations. If a particle's fitness does not change for some epochs, its position is set to be a candidate solution. Then, this particle is removed from the main swarm and a new sub-swarm is generated. As the algorithm proceeds, the main swarm loses its members as the new sub-swarms are created. Dynamically generated sub-swarms are expected to locate all global and local optima in parallel.

NichePSO

Initialize particles in the main swarm

Repeat

1. Train particles in the main swarm using a single iteration of the cognition only model
2. Update fitness of each particle in the main swarm
3. For each subswarm:
 - a. Train subswarm particles using a single iteration of the GCPSO algorithm
 - b. Update fitness of each particle
 - c. Update subswarm radius
4. If possible, merge subswarms
5. Allow subswarms to absorb any particles from the main swarm that moved into it
6. Search main swarm for any particle that meets the partitioning criteria – If found, create a new subswarm with this particle and its closest neighbor

Until stopping criteria are met

Fig. 1. Pseudocode for the NichePSO algorithm

The algorithm of Løvbjerg et al. [10] is adapted for improving swarm diversity. If the swarm size is small, then PSO algorithm has a disadvantage of getting stuck at a position when $x_i \cong y_i \cong y_g$, where velocity might approach to zero. Therefore, NichePSO uses Van den Bergh's GCPSO algorithm [1] to prevent sub-swarms from halting.

The initial swarm is vital for the success of the NichePSO, hence Faure-sequences are used to distribute the initial particles uniformly over the search space. If a particle does not belong to a niche, and the variance of its fitness is below a threshold for some epochs, then a niche is created around it. These niches may merge, if the distance between the best particles in them is less than some value μ or absorb the other particles which do not belong to a niche. NichePSO is implemented as described in [5]. It is observed that the niche radius may increase, spanning the whole search space and causing most of the particles to converge to a single optimum. In this paper, a modified version, referred as *mNichePSO* is proposed to prevent this type of behavior. Simply, the niche radius size is not allowed to exceed a maximum value.

3 PSO with Crazyness and Hill Climbing (CPSO)

In most of the algorithms used for optimization, the balance between exploration and exploitation is vital for success. The proposed CPSO algorithm (Fig. 2) for multimodal function optimization uses a random walk component and a hill climber to enhance the exploration and exploitation capabilities of PSO, respectively.

CPSO

Initialize particles

Repeat

1. On the first and every m epochs, construct sub-swarms according to their geographical positions, with a neighborhood size n .
2. For each particle compute 2 candidate positions:
 - a. Use the original PSO, where y_g is the sub-swarm's best.
 - b. if (the fitness variance of a particle or particles in a subswarm for k epochs is smaller than a *variance* threshold)
 - then
 - use the original PSO, where y_g is the sub-swarm's best
 - else
 - generate a random position using (5)
3. Compute the fitness values of these candidate positions. Choose the position with the better fitness as particles' current positions. If the random position produces a better fitness, set the particles' velocities using (6).
4. If (the fitness variance of a particle for p epochs is smaller than a *variance* threshold)
 - then
 - reset the velocity of the particle using (6).

Until stopping criteria met

Fig. 2. Pseudocode for the CPSO algorithm

In this algorithm, the main swarm is divided into sub-swarms of size n according to their geographical positions. Starting from the first particle, for each particle which does not belong to a sub-swarm, the nearest $(n-1)$ particles, which also do not belong to a sub-swarm, are detected. At every m epochs, the sub-swarms are rearranged according to their current geographical positions. This action provides a type of

communication and information diffusion between particles, since a local best value might change within a neighborhood. Each particle generates two candidate positions, denoted by x^1 and x^2 , at each epoch. Let v^1 and v^2 denote velocities for the related candidate positions. In (3), y_{gi} is the best position visited so far within the neighborhood of the i^{th} particle. The parameters $maxx_d$ and $minx_d$, in (5), are the limits of the search space at dimension d . The first candidate position is computed using (3) and (4), and the second one is computed using (5) during the initial moves. For the candidate positions x^1 and x^2 , each particle makes a decision based on the fitness values. A particle moves to the position that generates a better fitness.

$$v_{id}^1(t+1) = w v_{id}^1(t) + c_1 r_1(t) (y_{id}(t) - x_{id}(t)) + c_2 r_2(t) (y_{gid}(t) - x_{id}(t)), \quad (3)$$

$$x_{id}^1(t+1) = x_{id}^1(t) + v_{id}^1(t+1), \quad (4)$$

$$x_{id}^2(t+1) = minx_d + r_3 (maxx_d - minx_d). \quad (5)$$

Moving to a better candidate position can be considered as a hill climbing step. A single meme consisting of two phases is used: *sampling* and *acceptance*. As a sampling technique either a random walk (craziness) or the PSO algorithm itself is invoked, depending on the mode of operation as described in Fig. 2. As an acceptance strategy, only improving moves are admitted. If the position x^2 is chosen, then the previous velocity becomes useless. Hence, a new velocity has to be assigned to the particle. Equation (6) is used for that purpose.

$$v_{id}(t+1) = \alpha maxv r_5(t+1), \quad (6)$$

where α is a constant number, $maxv$ is the limit for the velocity of the particles, and $r_5(t)$ is a uniform random number in $[-1, 1]$ at time t . If the *variance* of the fitness for the last p epochs is smaller than a threshold value, all particles in a sub-swarm stop making random moves. The mode of operation switches to a refined search. Particles generate two velocities invoking the (3) twice, yielding two candidate positions and the search continues as described in Fig. 2. CPSO algorithm introduces the following parameters: n , k , p , m , *variance* (threshold) and α .

4 Experiments

The runs are performed on a 2 GHz, Windows 2003 operating system with 512 MB of memory. A Matlab application is implemented for the experiments, available at <http://cse.yeditepe.edu.tr/ARTI/projects/cpso>. Each experiment is repeated 50 times. A run is terminated either the maximum number of evaluations is exceeded or all required global optima are found within a fitness range of 0.00005.

4.1 Experimental Setup and Comparison Criteria

Well-known benchmark functions are used during the experiments as presented in Table 1. The initial experiments are performed for obtaining the best set of parameters

for CPSO. Then, seven different multimodal PSO algorithms (SPSO, NichePSO, nbestPSO, CPSO, UPSO, mNichePSO, PVPSO) are tested on 10 different benchmark functions (F1-F10). The swarm size is chosen as 30 and 50, for functions F1-F4 and F5-F10, respectively. The problem dimension is one for the functions F1-F4, two for the functions F5-F10. For a fair comparison, the maximum number of evaluations is limited to 15,000 for F1-F4, and 25,000 for F5-F10. In UPSO, X is used as 0.729, and c_1 and c_2 are set to 2.05. For the rest of the algorithms, the inertia weight is linearly decreased from 0.8 to 0.6, and c_1 and c_2 are set to 1.5 for a stable PSO. More discussions on the choice of parameters can be found in [6], [8], [11] and [12].

Table 1. Benchmark functions used during the experiments; *minx* and *maxx* indicate the lower and upper bound for each dimension, #gl. and #lo. indicate the total number of global and local optima, respectively, for the corresponding function

| lb. | Function | <i>minx, maxx</i> | #gl. | #lo. | Source(s) |
|-----|--|-------------------|------|------|-----------|
| F1 | $y=1-\sin(5\pi x)^6$ | 0.0, 1.0 | 5 | 0 | [8,5,1] |
| F2 | $y=1-\exp(-2\log(2)((x-0.1)/0.8)^2)\sin(5\pi x)^6$ | 0.0, 1.0 | 1 | 4 | [8,5] |
| F3 | $y=1-\sin(5\pi(x^{(3/4)}-0.05))^6$ | 0.0, 1.0 | 5 | 0 | [8,5,1] |
| F4 | $y=1-(\exp(-2\log(2)*((x-0.08)/0.854)^2)\sin(5\pi(x^{(3/4)}-0.05))^6)$ | 0.0, 1.0 | 1 | 4 | [8,5,1] |
| F5 | $z=(x^2+y-11)^2-(x+y^2-7)^2$ | -10.0, 10.0 | 4 | 0 | [8,5] |
| F6 | $z=\sin(2.2\pi x+\pi/2)((2- y)/2)((3- x)/2)+\sin(0.5\pi y^2+\pi/2)((2- y)/2)((2- x)/2)$ | -2.0, 2.0 | 4 | 8 | [16,17] |
| F7 | $z=\cos(x)^2+\sin(y)^2$ | -4.0, 4.0 | 6 | 0 | [12] |
| F8 | $z=(\cos(2x+1)+2\cos(3x+2)+3\cos(4x+3)+4\cos(5x+4)+5\cos(6x+5))(\cos(2y+1)+2\cos(3y+2)+3\cos(4y+3)+4\cos(5y+4)+5\cos(6y+5))$ | -2.0, 2.5 | 2 | 38 | [8,15] |
| F9 | $z=(y^2-4.5y^2)xy-4.7\cos(3x-y^2(2+x))\sin(2.5\pi x)+(0.3x)^2$ | -1.2, 1.2 | 1 | 9 | [13] |
| F10 | $z=4x^2-2.1x^4+(1/3)x^6+xy-4y^2+4y^4$ | -1.9, 1.9 | 2 | 4 | [13] |

In SPSO, the species radius is chosen as $(\max x-\min x)/20$. In NichePSO, the parameters μ and the variance are set to 0.001 and 0.0001, respectively [5]. Additionally, in mNichePSO, the maximum niche radius is set to $(\max x-\min x)/10$. In nbestPSO, the neighborhood size is linearly decreased from 6 to 2. To evaluate the effect of the maximum velocity, each algorithm is investigated with two different values: $(\max x-\min x)/20$ and $(\max x-\min x)/2$, where each algorithm is labeled as *algorithm_abbreviation-20* and *algorithm_abbreviation-2*, respectively.

In the multimodal optimization problems, not only multiple global optima having equal quality are to be searched, but also a predetermined set of local optima might be required. Therefore, it is difficult to evaluate and compare different algorithms. *Global peak ratio (gpr)* is defined as the ratio of the average number of global optima found to the total number of global optima. *Local peak ratio (lpr)* is defined similarly

using the local optima. *Global success consistency ratio (gscr)* denotes the proportion of the runs in which all global optima are discovered to the total number of runs, and *local success consistency ratio (lscr)* denotes the proportion of the runs in which all local are discovered to the total number of runs. The *overall success rate (osr)* is used as a single comparison criterion to evaluate all these aspects at once as shown in (7). If there are no local optima, *lpr* and *lscr* are set to 0, and the divisor to 2. Consequently, *osr* values are in the range [0,1] and a higher ratio indicates a better per-formance

$$osr = \frac{(gpr + gscr + lpr + lscr)}{4} . \tag{7}$$

4.2 Parameter Tuning for CPSO

324 different parameter sets are tested based on the combination of the following values for each CPSO parameter with $maxv=(maxx-minx)/20$:

- The sub-swarm size; $n \in \{2, 3, 4\}$
- The number of epochs $k \in \{3, 5, 7\}$ (step 2 (b) in Fig. 2)
- The number of epochs $p \in \{3, 5, 7\}$ (step 4 in Fig. 2)
- The *variance* (threshold) $\in \{0.0001, 0.01\}$ (step 2 (b) and 4 in Fig. 2)
- The number of epochs $m \in \{5, 7, 10\}$ for the sub-swarm reconstruction
- The velocity reset constant $\alpha \in \{0.001, 0.01\}$ (step 3 and 4 in Fig. 2, (6))

The performance of CPSO with each parameter set is compared with respect to the average *osr* over all benchmark functions. The results are summarized in Table 2. The best parameter set contains: $n=2, k=3, p=3, variance=0.01, m=5$ and $\alpha=0.01$. It seems that relatively low values for $n, k, p,$ and m and relatively high values for *variance* and α are *good* initial choices in a CPSO.

Table 2. The parameter sets that rank top ten based on their performances

| <i>rank</i> | <i>variance</i> | <i>n</i> | <i>k</i> | <i>p</i> | <i>m</i> | α |
|-------------|-----------------|----------|----------|----------|----------|----------|
| 1 | 0.01 | 2 | 3 | 3 | 5 | 0.01 |
| 2 | 0.01 | 2 | 3 | 5 | 7 | 0.01 |
| 3 | 0.01 | 2 | 3 | 3 | 7 | 0.01 |
| 4 | 0.01 | 2 | 3 | 7 | 5 | 0.01 |
| 5 | 0.01 | 2 | 3 | 5 | 5 | 0.01 |
| 6 | 0.01 | 2 | 3 | 5 | 5 | 0.001 |
| 7 | 0.01 | 2 | 3 | 7 | 10 | 0.01 |
| 8 | 0.01 | 2 | 3 | 3 | 10 | 0.01 |
| 9 | 0.01 | 2 | 3 | 7 | 7 | 0.01 |
| 10 | 0.01 | 2 | 3 | 3 | 5 | 0.001 |

4.3 Experimental Results

During the experiments, none of the algorithms is capable of finding all optima on all functions as summarized in Table 3. The maximum velocity choice can affect the performance of an algorithm considerably. NichePSO, mNichePSO, CPSO and PVPSO are more sensitive to the maximum velocity as compared to the others. On average, NichePSO, mNichePSO, CPSO algorithms perform better with a relatively low maximum velocity. PVPSO is the only algorithm that performs better with a relatively high maximum velocity. This choice does not generate a significant performance variance for SPSO, nbestPSO and UPSO.

Table 3. Average *osr* of each algorithm over all runs for each benchmark function. The best values (algorithms) for the benchmark functions are marked with the bold entries.

| Algorithm | F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 | F10 | avr.osr. | stdev. |
|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|--------|
| SPSO-2 | 0.98 | 0.82 | 0.96 | 0.84 | 1.00 | 0.64 | 0.97 | 0.58 | 0.65 | 0.59 | 0.80 | 0.17 |
| SPSO-20 | 0.99 | 0.99 | 1.00 | 0.94 | 0.95 | 0.48 | 0.72 | 0.27 | 0.64 | 0.82 | 0.78 | 0.25 |
| NichePSO-2 | 0.45 | 0.69 | 0.40 | 0.64 | 0.13 | 0.10 | 0.15 | 0.13 | 0.59 | 0.14 | 0.34 | 0.24 |
| NichePSO-20 | 0.97 | 1.00 | 0.94 | 0.96 | 0.19 | 0.21 | 0.45 | 0.13 | 0.66 | 0.45 | 0.60 | 0.35 |
| nbest-2 | 0.71 | 0.85 | 0.77 | 0.80 | 0.89 | 0.47 | 0.72 | 0.41 | 0.52 | 0.60 | 0.67 | 0.17 |
| nbest-20 | 0.81 | 0.93 | 0.74 | 0.84 | 0.90 | 0.43 | 0.74 | 0.31 | 0.46 | 0.58 | 0.67 | 0.22 |
| CPSO-2 | 0.96 | 0.78 | 0.83 | 0.78 | 0.28 | 0.56 | 0.98 | 0.17 | 0.56 | 0.53 | 0.64 | 0.27 |
| CPSO-20 | 0.96 | 0.87 | 0.90 | 0.90 | 1.00 | 0.66 | 1.00 | 0.52 | 0.63 | 0.62 | 0.81 | 0.18 |
| UPSO-2 | 0.72 | 0.70 | 0.66 | 0.70 | 0.81 | 0.47 | 0.64 | 0.51 | 0.56 | 0.50 | 0.63 | 0.11 |
| UPSO-20 | 0.72 | 0.81 | 0.68 | 0.74 | 0.88 | 0.54 | 0.55 | 0.48 | 0.45 | 0.54 | 0.64 | 0.15 |
| mNiche-2 | 0.70 | 0.69 | 0.61 | 0.67 | 1.00 | 0.54 | 0.91 | 0.62 | 0.75 | 0.64 | 0.71 | 0.14 |
| mNiche-20 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0.66 | 0.99 | 0.14 | 0.47 | 0.91 | 0.82 | 0.30 |
| PVPSO-2 | 0.99 | 0.93 | 0.98 | 0.97 | 0.45 | 0.57 | 0.79 | 0.39 | 0.55 | 0.62 | 0.72 | 0.23 |
| PVPSO-20 | 1.00 | 1.00 | 1.00 | 1.00 | 0.21 | 0.38 | 0.46 | 0.20 | 0.35 | 0.71 | 0.63 | 0.35 |

The proposed modification to NichePSO, mNichePSO as described in Sect. 2.1, delivers a significantly better performance compared to the original one considering the average *osr*. Furthermore, mNiche-20, CPSO-20 and SPSO-2 are the top three algorithms in that order with respect to the average *osr*. PVPSO-20 is as successful as mNichePSO-20 for locating the multiple optima in the 2D search landscapes, while its performance deteriorates extremely for the 3D search landscapes.

The particle positions at the end of a sample run of a PSO algorithm for F6 are illustrated in Fig. 3. SPSO is good at locating the required optima. Mostly, at end of the runs, the species were evenly distributed around these optima. NichePSO provides good separation of niches, but, sometimes, all niches might merge into a single one. Preventing the niche radius to grow, as in mNichePSO, yields a better diversity. The nbestPSO produces a poor convergence rate. The proposed CPSO algorithm is very successful in locating the global optima as compared to the local optima (Fig. 3-4). Although, UPSO performs poorly, the parameters provide an effective way to derive the algorithm to locate any optima.

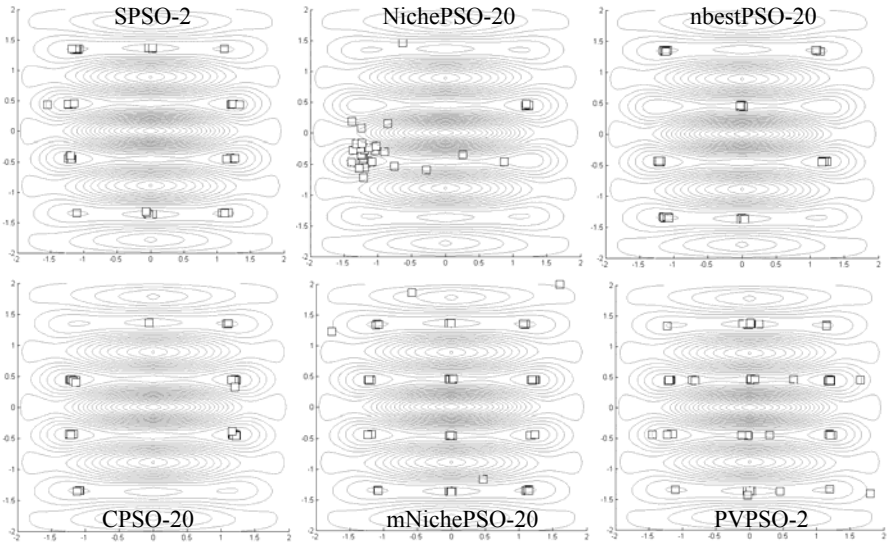


Fig. 3. Particle positions after a sample run of each algorithm on F6

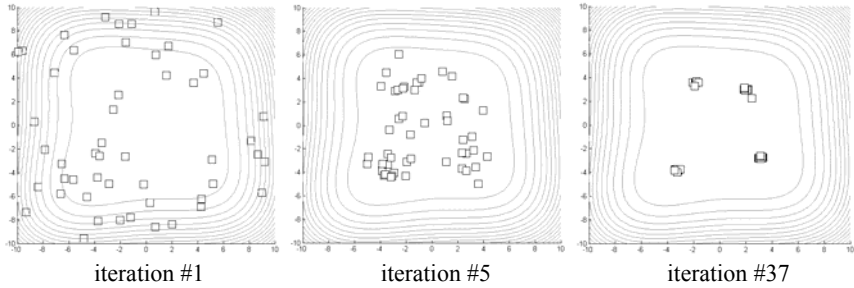


Fig. 4. A sample run of CPSO-20 on F5

5 Conclusions

There are five essential PSO algorithms that have been proposed for multimodal function optimization: SPSO, NichePSO, nbestPSO, UPSO and PVPSO. During the preliminary experiments, it is observed that the performance of NichePSO can be improved significantly restricting the growth of the niche radius beyond a predefined value. mNichePSO performs much better than NichePSO on the benchmark functions, considering the evaluation criteria. In this paper, a new multimodal particle swarm optimization algorithm called CPSO is introduced. CPSO is compared against these approaches. All algorithms introduce additional parameters, requiring fine tuning. Except UPSO, all algorithms are computationally expensive due to the need of the distance calculations. If the dimensionality increases, the scalability issue might arise, causing this cost to become remarkable. In CPSO, the cost is some what

reduced by making the computations at a predefined frequency. With its enhanced exploration and exploitation capabilities based on craziness and hill climbing, CPSO has a good performance especially in locating multiple global optima, matching the overall performance of mNichePSO and SPSO on the benchmark functions.

References

1. D. Beasley, D.R. Bull, R.R. Martin, A Sequential Niching Technique for Multimodal Function Optimization, *Evolutionary Computation*, 1(2), MIT Press, 1993, pp. 101-125.
2. F. van den Bergh, An Analysis of Particle Swarm Optimizers, PhD Thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa, 2002.
3. F. van den Bergh, A.P. Engelbrecht, A study of particle swarm optimization particle trajectories, *Information Sciences* 176, 2006, pp. 937-971.
4. R. Brits, A.P. Engelbrecht, F. vanden Bergh, Solving Systems of Unconstrained Equations using Particle Swarm Optimization, *Int. Conf. on Sys., Man and Cyber.*, v.3, 2002, no.pp. 6.
5. R. Brits, A.P. Engelbrecht, F. van den Bergh, A niching particle swarm optimizer, *Proc. 4th Asia-Pacific Conf. on Simulated Evolution and Learning*, vol. 2, 2002, pp. 692-696.
6. R. C. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, *Proc. of the IEEE Congress on Evolutionary Comp.*, 2000, pp. 84-88.
7. J. Kennedy, R.C. Eberhart, Particle Swarm Optimization, *Proc. IEEE Int. Conf. on N.N.*, 1995, pp. 1942-1948.
8. J.-P. Li, M.E. Balazs, G.T. Parks, P.J. Clarkson, A Genetic Algorithm using Species Conservation for Multimodal Function Optimization, *Journal of Evolutionary Computation*, vol. 10, no. 3, 2002, pp. 207-234.
9. X. Li, Adaptively Choosing Neighborhood Bests using Species in a Particle Swarm Optimizer for Multimodal Function Optimization, *Proc. of GECCO 2004*, LNCS 3102, eds. Deb, K. et al., Springer-Verlag, Seattle, USA, 2004, pp. 105-116.
10. M. Løvbjerg, T.K. Rasmussen, T. Krink, Hybrid Particle Swarm Optimizer with Breeding and Subpopulations, *Proc. of the Genetic and Evolutionary Comp. Conf.*, v. 1, 2001, pp. 469-476.
11. E. Ozcan, C.K. Mohan, Particle Swarm Optimization: Surfing the Waves, *Proc. of IEEE Congress on Evolutionary Computation*, Piscataway, NJ. 1999, pp. 1939-1944.
12. K. E. Parsopoulos, M. N. Vrahatis, Modification of the particle swarm optimizer for locating all the global minima, *Proc. of the ICANNGA*, 2001, pp. 324-327.
13. K. E. Parsopoulos, M. N. Vrahatis, UPSO: A Unified Particle Swarm Optimization Scheme, *Lecture Series on Comp. and Computational Sci.*, Vol. 1, *Proc. of the Int. Conf. of Computational Methods in Sci. and Eng.*, 2004, pp. 868-873.
14. I.L. Schoeman, A.P. Engelbrecht, A Parallel Vector-Based Particle Swarm Optimizer, *Proc. of the International Conf. on Neural Networks and Genetic Algorithms*, 2005, pp. 268-271.
15. D.G. Sotiropoulos, V.P. Plagianakos, M.N. Vrahatis, An evolutionary algorithm for minimizing multimodal functions, *Proc. of the Fifth Hellenic- European Conf. on Comp. Math. and its App.*, vol. 2, 2002, pp. 496-500.
16. F. Streichert, G. Stein, H. Ulmer, A. Zell, A Clustering Based Niching EA for Multimodal Search Spaces, *Artificial Evolution*, 2003, pp. 293-304.
17. R.K. Ursem, Multinational evolutionary algorithms, *Proc. of the 1999 Congress of Evolutionary Computation (CEC-1999)*, vol. 3, 1999, pp. 1633-1640.

Quantum-Behaved Particle Swarm Optimization with Binary Encoding

Jun Sun, Wenbo Xu, Wei Fang, and Zhilei Chai

Center of Intelligent and High Performance Computing,
School of Information Technology, Southern Yangtze University,
No. 1800, Lihudadao Road, Wuxi,
214122 Jiangsu, China

{sunjun_wx, xwb_sytu, wxfangwei}@hotmail.com, zlchai@gmail.com

Abstract. The purpose of this paper is to generalize Quantum-behaved Particle Swarm Optimization (QPSO) Algorithm to discrete binary search space. To design Binary QPSO (BQPSO), we redefine the position vector and the distance between two positions, and adjust the iterative equations of QPSO to binary search space. The operations designed for BQPSO are far different from those in BPSO, but somewhat like those in Genetic Algorithms (GAs). Therefore, BQPSO integrates strongpoint of GA with the features of PSO, which make it able to find out the global optimum of the problem more efficiently than BPSO, as shown by the experiment results of BQPSO and BPSO on De Jong's five test functions.

1 Introduction

The Particle Swarm Optimization (PSO) algorithm, firstly proposed by Kennedy and Eberhart [4], is a population-based evolutionary search technique. Its underlying motivation for the development of PSO was social behavior of animals such as bird flocking, fish schooling, and animal herding and swarm theory. In PSO with M individuals, a potential solution to a problem is represented as a particle flying in D -dimensional search space, with the position vector $X_i = (X_{i1}, X_{i2}, \dots, X_{iD})$ and velocity $V_i = (V_{i1}, V_{i2}, \dots, V_{iD})$. Each particle records its best previous position (the position giving the best fitness value) as $pbest_i = (pbest_{i1}, pbest_{i2}, \dots, pbest_{iD})$ called personal best position. At each iteration, each particle competes with the others in the neighborhood or in the whole population for the best particle (with best fitness value among neighborhood or the population) with best position $gbest_i = (gbest_{i1}, gbest_{i2}, \dots, gbest_{iD})$ called global best position, and then makes stochastic adjustment according to the following evolution equations.

$$V_{id} = w \cdot V_{id} + c_1 \cdot rand(\cdot) \cdot (pbest_{id} - X_{id}) + c_2 \cdot Rand(\cdot) \cdot (gbest_d - X_{id}) \quad (1)$$

$$X_{id} = X_{id} + V_{id} \quad (2)$$

for $i = 1, 2, \dots, M$; $d = 1, 2, \dots, D$. In the above equations, c_1 and c_2 are positive constant; $\text{rand}()$ and $\text{Rand}()$ are two random functions generating uniformly distributed random numbers within $[0, 1]$. Parameter w is the inertia weight introduced to accelerate the convergence speed of the PSO. At each iteration, the value of V_{id} is restricted in $[-V_{max}, V_{max}]$.

In original PSO, the particles operate in continuous search space, where the trajectories are defined as changes in positions. In discrete binary PSO [6], trajectories are defined as changes in the probability that a coordinate of the position vector will take on a value from feasible discrete values.

Recently, Sun *et al* developed a novel variant of PSO [8], [9]. It was named as Quantum-behaved Particle Swarm Optimization (QPSO), since the new PSO is inspired by the quantum theory. The results of experiments on several widely known benchmark functions show that QPSO outperforms PSO in search abilities and seems to be a promising optimization problem solver.

This paper will focus on developing a discrete binary version of QPSO (BQPSO). Because QPSO operates in accord with a set of iterative equations far different from equation (1) and (2), the methodology of BPSO does not apply to BQPSO and a new set of operators for BQPSO will be proposed. The rest part of the paper is organized as follows. In Section 2, a brief introduction of BPSO is presented. The concepts of QPSO are described in Section 3. And Section 4 presents the proposed BQPSO. Section 5 gives the experiment results of BQPSO and BPSO on De Jong's suite of five test functions. The paper is concluded in Section 5.

2 Discrete Binary PSO

Kennedy and Eberhart proposed the original PSO for the use in continuous search space. Many Optimization problems, however, are set in a space featuring discrete, qualitative distinctions between variables and between levels of variables. For this reason, Kennedy and Eberhart modified equations in continuous PSO to be adapted to discrete binary search space. In their proposed binary version of the PSO (BPSO), trajectories, velocities, etc., are defined in terms of probabilities that a bit will be one or the other. Thus, a particle moves in a state space restricted to 0 or 1 on each dimension (each bit), where each V_{id} represents the probability of bit X_{id} taking the value 1 and updates according to equation (1). The main difference between original PSO and the BPSO is that equation (3) replaces equation (2).

$$\text{if } (\text{rand}() < S(V_{id})) \text{ then } X_{id} = 1 \text{ else } X_{id} = 0 \quad (3)$$

where $S(v)$ is a sigmoid limiting transformation function ($S(v) = 1/(1+e^{-v})$), and $\text{rand}()$ is a random number selected from a uniform distribution in $[0, 1]$.

In the discrete binary version, V_{max} is retained, that is $|V_{id}| < V_{max}$, which simply limits the ultimate probability that bit X_{id} will take on a binary value. From equation (3), we can infer that a smaller V_{max} will allow a higher mutation rate.

3 QPSO Algorithm

In Quantum-behaved Particle Swarm Optimization (QPSO), the particle moves according to the following equation

$$mbest = \sum_{i=1}^M pbest_i / M =$$

$$= \left(\sum_{i=1}^M pbest_{i1} / M, \sum_{i=1}^M pbest_{i2} / M, \dots, \sum_{i=1}^M pbest_{id} / M \right) \quad (4)$$

$$p_{id} = \varphi * pbest_{id} + (1 - \varphi) * gbest_d, \quad \varphi = rand() \quad (5)$$

$$X_{id} = p_{id} \pm \alpha * |mbest_d - X_{id}| * \ln(1/u), \quad u = Rand() \quad (6)$$

where

mbest The mean best position among the particles, which is calculated by equation (4).

p_{id} A stochastic point between $pbest_{id}$ and $gbest_d$, which the d th coordinate of the local attractor of the i th particle, P_i

φ A random number distributed uniformly on [0,1],

u Another random number distributed uniformly on [0,1],

α The parameter of QPSO, called Contraction-Expansion Coefficient.

The procedure of Quantum-behaved Particle Swarm Optimization (QPSO) Algorithm in [9] is outlined as follows:

QPSO Algorithm

Initialize particles with random position $X_i = X[i][:]$;

Let personal best position $pbest_i = pbest[i][:] = X_i$;

while termination criterion is not met **do**

 Compute the mean best position *mbest* by equation (4);

for $i=1$ to swarm size M

if $f(X_i) < f(pbest_i)$ **then** $pbest_i = X_i$; **endif**

 Find the $gbest = pbest[g][:]$ across the swarm;

for $d=1$ to D

$fi = rand(0, 1)$; $u = rand(0, 1)$;

$pid = fi * pbest[i][d] + (1 - fi) * gbest[d]$;

if $(rand(0, 1) > 0.5)$

$X[i][d] = pid + \alpha * abs(mbest[d] - X[i][d]) * \ln(1/u)$;

else

$X[i][d] = pid - \alpha * abs(mbest[d] - X[i][d]) * \ln(1/u)$;

endif

endfor

endfor

endwhile

The iterative equations (4)~(6) are derived from a quantum δ potential well model proposed in [8]. For the detailed information about QPSO, one can refer to the literature such as [8] and [9].

4 Discrete Binary QPSO

In this section, a discrete binary version of QPSO (BQPSO) is proposed. Because the iteration equations of QPSO is far different from those of PSO, the methodology of BPSO does not apply to QPSO. In QPSO, there are no velocities and trajectories concepts but position and distance. Since position of the particle is represented as a binary string, the key problem of designing the BQPSO is how to define the distance between two positions and the transformation.

In our proposed BQPSO, the distance is defined as the Hamming distance between two binary strings. That is

$$|X - Y| = d_H(X, Y) \quad (7)$$

where X and Y are two binary strings, say two position. The function $d_H(\cdot)$ is to get the Hamming distance between X and Y . In comparing two bit strings, the Hamming distance is the count of bits different in the two strings.

It is necessary to emphasize that the dimension in BQPSO is defined as the number of decision variables like that in continuous PSO, instead of as the length of the binary string like in BPSO. In BQPSO, the variable with subscripts i and d represents a substring in the binary string, that is, X_{id} refers to the d th substring (d th decision variable) of the position of the i th particle, not the d th bit in a binary string. Given that the lengths of X_{id} and X_i are l_d and l respectively, we can obtain

$$l = \sum_{i=1}^D l_d, \quad d = 1, 2, \dots, D \quad (8)$$

The key of BQPSO design is to adjust the evolution equations (4)~(6) to discrete binary space. In continuous QPSO, the mean best (*mbest*) position of all particles is calculated by equation (4), whereas in our proposed BQPSO, the j th bit of the *mbest* is determined by the states of the j th bits of all particles' *pbests*. If more particles take on 1 at the j th bit of their own *pbests*, the j th bits of *mbest* will be 1; otherwise the bit will be 0. However, if half of the particles take on 1 at the j th bit of their *pbests*, the j th bit of *mbest* will be set randomly to be 1 or 0, with probability 0.5 for either state. The pseudo-code of the function for obtaining *mbest* is given as follows.

```

Get_mbest (pbest)
for  $j=1$  to  $l$  (the length of binary string)
    sum=0;
    for each particle  $i$ 
        sum=sum+pbest [ $i$ ] [ $j$ ];
    endfor
    avg=sum/M;

```

```

if avg>0.5  mbest[j]=1;endif
if avg<0.5  mbest[j]=0;endif
  if avg=0.5
    if rand( )<0.5 mbest[j]=0;
      else mbest[j]=1;
    endif
  endif
endfor
Return mbest

```

In the above pseudo-code, the inputs of the function Get_mbest() is binary strings of all particles' personal best positions (pbest) and output is the string of *mbest*.

The equation (5) is used for getting the coordinate of the local attractor P_i for particle i . In continuous QPSO, the coordinate p_{id} of P_i lies between $pbest_{id}$ and $gbest_{id}$. Therefore $P_i = (p_{i1}, p_{i2}, \dots, p_{id})$ distributes randomly in the hyper-rectangle with $pbest_i$ and $gbest$ being its two diagonal ends. The distance of P_i to either $pbest_i$ or $gbest$ must be less than the length of the diagonal line. That is

$$|P_i - pbest_i| \leq |pbest_i - gbest| \text{ and } |P_i - gbest| \leq |pbest_i - gbest| \tag{9}$$

Hence, we can infer that the particle's converging to point P_i reduces the diversity of the population, which corresponds to the local search (exploitation) of the particle. In BQPSO, the point P_i is generated through crossover operation like that used in Genetic Algorithm (GA). That is, P_i is randomly selected from two offspring that are generated by exerting crossover on the two parents, $pbest_i$ and $gbest$. Obviously, The position of P_i resulted from crossover satisfies the equation (9) in terms of Hamming distance in binary space. It is logical that in BQPSO, the point P_i is obtained by one-point or multi-point crossover operation, because the purpose of crossover is to reduce the diversity and undertake local search. Outlined below is the procedure corresponding to equation (5).

```

Get_P(pbesti, gbest)
  Exert crossover operation on pbesti and gbest to
  generate two offspring binary strings z1 and z2;
  if rand( )<0.5
    Pi=z1
  else Pi=z2;
  endif
Return Pi

```

Consider iterative equation (6) and transform it as

$$|X_{id} - p_{id}| = \alpha |mbest_d - X_{id}| \ln(1/u), \quad u = Rand() \tag{10}$$

In terms of Hamming distance, the equation can be written as

$$d_H(X_{id}, p_{id}) = \lceil b \rceil \tag{11}$$

$$b = \alpha * d_H(X_{id}, mbest_d) * \ln(1/u), \quad u = Rand() \tag{12}$$

where $d_H(X_{id}, p_{id})$ is the Hamming distance between substring X_{id} of the new position and the d th substring p_{id} in P_i . $mbest_d$ is the d th substring in string of $mbest$. The reason for the use of function $\lceil \cdot \rceil$ is that the Hamming distance must be an integer. According to the above equations, we can obtain the new substring X_{id} by the procedure of inverting the states of $\lceil b \rceil$ randomly selected bits in substring p_{id} , which, however, has time complexity $O(b \cdot l_d)$ at worst. To reduce the computation cost, we replace the procedure by the transformation in which each bit in p_{id} is mutated with the probability computed by

$$Pr = \begin{cases} b / l_d \\ 1 \end{cases} \quad \text{if } b / l_d > 1 \quad (13)$$

The transformation is described as follows.

```

Transf ( $p_{id}$ ,  $Pr$ )
for each bit in the substring  $p_{id}$ ;
    if rand() <  $Pr$ 
        if the state of the bit is 1
            Set its state to 0;
        else set its state to 1;
        endif
    endif
endfor
 $X_{id} = p_{id}$ ;
Return  $X_{id}$ 

```

The function $\text{Transf}()$ has time complexity $O(l_d)$. It operates on each substring of a position, instead of on the whole binary string of the position as $\text{Get_mbest}()$ and $\text{Get_P}()$ do.

With the above definition and modifications of iterative equations, the Binary Quantum-behaved Particle Swarm Optimization (BQPSO) algorithm can be described as the following procedure:

- Step 1.** Initialize an array of binary bits for all particles: X and $pbest = X$;
- Step 2.** Determine the mean best position among the particles by $\text{Get_mbest}(pbest)$;
- Step 3.** Evaluate the desired objective function (minimization problem for example) for each particle and compare with the particle's previous best values: if $f(X_i) < f(pbest_i)$, then $pbest_i = X_i$.
- Step 4.** Determine the current global position minimum among the particle's best positions. That is: $g = \arg \min_{1 \leq i \leq M} (f(P_i))$ (M is the population size). Thus, $gbest = pbest_g$.
- Step 5.** Compare the current global position $gbest$ to the previous global position: if the fitness value of current global position is less than that of the previous global position, then set the global position to the current global position.
- Step 6.** For each particle, get a stochastic position P_i by $P_i = \text{Get_P}(pbest_i, gbest)$.
- Step 7.** For each dimension (each substring corresponding to dimension d), compute the mutation probability Pr for substring P_{id} by (12) and (13).

Step 8. Generate the new substring X_{id} by Transf (P_{id}, Pr). And get the new position X_i by combining all new substring $X_{id}(d=1,2,\dots, D)$

Step 9. Repeat steps (2) –(8) until a stop criterion is satisfied OR a pre-specified number of iterations are completed

5 Experiment Results

The performance of the proposed BQPSO algorithm was tested on the following five different standard functions to be maximized. These functions were designed by De Jong to test a carefully constructed set of dimensions of performance of an optimizer [3].

$$F_1(X) = 78.6 - \sum_{i=1}^3 x_i^2, \quad (-5.12 \leq x_i \leq 5.12), \tag{14}$$

$$F_2(X) = 3905.93 - (100(x_1^2 - x_2)^2 - (1 - x_1)^2), \quad (-2.048 \leq x_i \leq 2.048) \tag{15}$$

$$F_3(X) = 25 - (x_1 + x_2 + x_3 + x_4 + x_5), \quad x_i \in Z, \quad (-5.12 \leq x_i \leq 5.12) \tag{16}$$

$$F_4(X) = 12482 - \sum_{i=1}^{30} x_i^4, \quad (-1.28 \leq x_i \leq 1.28) \tag{17}$$

$$F_5(X) = 500 - 1 / \left(0.002 + \sum_{j=1}^{25} \frac{1}{j+1 + \sum_{i=1}^2 (x_i - a_{ij})^6} \right), \tag{18}$$

$$a = \begin{pmatrix} 32.0 & 16.0 & 0 & 16.0 & 32.0 \\ -32.0 & -16.0 & 0 & 16.0 & 32.0 \end{pmatrix}, \quad (-65.536 \leq x_i \leq 65.536)$$

This section presents simulation results and performance comparison of BQPSO with BPSO, in terms of solution quality and the speed of finding the best solution. The functions were run 50 times each, with a population of 20 particles. The value parameters used for BPSO are the same as those specified in [6], i.e. V_{max} is set to 6; c_1 and c_2 equal to 1. The parameter α in BQPSO is set to 1.1 and 1.4 for two cases, respectively. The algorithm terminates when the number of iterations succeeds 200. We recorded the best fitness value (BFV) when the algorithm terminates at each run. The quality of the solutions produced by algorithms are evaluated by Mean Best Fitness Values (Mean BFV), Standard Deviation (St. Dev.), Maximum Best Fitness Value (Max. BFV) and Minimum Best Fitness Value (Min. BFV) out of 50 runs. These measurements are listed on Table 1.

Both BPSO and BQPSO were able to find out the optima of F1, whose fitness value is 78.6. BQPSO with either $\alpha=1.4$ or $\alpha=1.1$ hit the optima for 8 times out of 50 trial runs, whereas BPSO find out the optima for 6 times. As of solution quality, both

Table 1. Results of BQPSO and BPSO on five testing functions

| Function | Algorithm | Mean BFV | St. Deviation | Max. BFV | Min. BFV |
|----------------|------------|-------------|---------------|-------------|-------------|
| F ₁ | BPSO | 78.599846 | 8.38313E-05 | 78.6 | 78.5997 |
| | BQPSO(1.1) | 78.599864 | 8.51459E-05 | 78.6 | 78.5997 |
| | BQPSO(1.4) | 78.599871 | 8.43751E-05 | 78.6 | 78.5997 |
| F ₂ | BPSO | 3905.875171 | 0.199089262 | 3905.93 | 3904.927899 |
| | BQPSO(1.1) | 3905.894363 | 0.064367635 | 3905.93 | 3905.616996 |
| | BQPSO(1.4) | 3905.898361 | 0.110708752 | 3905.93 | 3905.153654 |
| F ₃ | BPSO | 54.94 | 0.239897937 | 55 | 54 |
| | BQPSO(1.1) | 54.94 | 0.239897937 | 55 | 54 |
| | BQPSO(1.4) | 54.94 | 0.239897937 | 55 | 54 |
| F ₄ | BPSO | 1252.669475 | 3.348750124 | 1258.628747 | 1245.186999 |
| | BQPSO(1.1) | 1252.008459 | 2.435689800 | 1256.59775 | 1247.783349 |
| | BQPSO(1.4) | 1251.612553 | 2.402303108 | 1258.540061 | 1245.607463 |
| F ₅ | BPSO | 498.6564274 | 0.470500681 | 499.2699104 | 497.7630575 |
| | BQPSO(1.1) | 498.7040881 | 0.457162494 | 499.2699104 | 497.7630575 |
| | BQPSO(1.4) | 498.7262385 | 0.395844667 | 499.2699104 | 498.1080945 |

BQPSO(1.4) and BQPSO(1.1) outperform BPSO in terms of Mean BFV and St. Deviation.

On the second F₂, both of BPSO and BQPSO were able to hit the optimum fitness value 39059300000. However BQPSOs generated better Mean BFV and St. Deviation, which means BQPSOs have better quality of solutions than BPSO.

The third function F₃ is an integer function with an optimum of 55. Both BQPSO and BPSO made 47 successful searches out of 50 runs.

Gaussian noise is introduced into F₄ function, which was measured as an average over the entire population, rather than a population best. The results show that BPSO generated slightly better Mean BFV but larger St. Deviation than BQPSO. On this function, BQPSO with $\alpha=1.1$ has better Mean BFV than BQPSO with $\alpha=1.4$.

The function F₅ has an optimum 500 and the algorithms are able to find out the best value 499.2699104. BQPSO are able to generate better and more stable solution than BPSO.

Figure 1 shows the convergence processes of BQPSO and BPSO on five testing functions. On F₁, BQPSO can converge to the optimum more rapidly than BPSO. On F₂, BPSO converges more quickly but generates worse solution. On F₃, BQPSO converges rapidly than BPSO at the early stage of running, but after about 50 iterations, BPSO catch up BQPSO and generates equal Mean BFV. On F₄, the convergence speed of BPSO is slower than BQPSO's. After 100 iterations, however, BPSO exceeds BQPSO and terminates at a slightly better solution. The convergence speed of BQPSO is faster than that of BPSO on F₅ obviously.

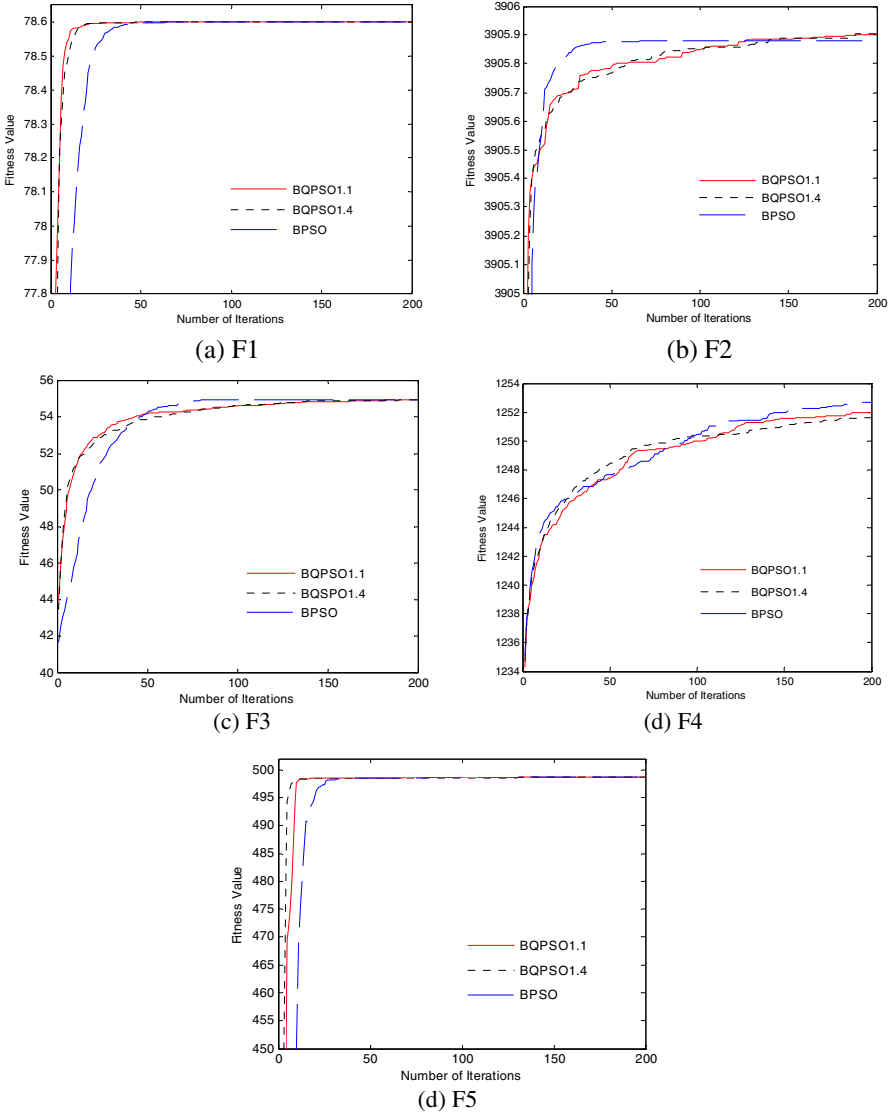


Fig. 1. The figure shows the convergence process of BQPSO and BPSO on five testing functions

6 Conclusions

In this paper, a discrete binary version of Quantum-behaved Particle Swarm Optimization algorithm (BQPSO) was proposed. The operation strategy of the BQPSO seems far different from that of the BPSO, but somewhat like that of the GA.

In BQPSO, the local attractor P_i is obtained by using the crossover operation on personal best position and global best position, and the transformation of getting new position X_i is similar to mutation operation in GA, which make BQPSO possess some features of GA. The experiment results show that BQPSO can find better solution generally than BPSO.

The BQPSO was implemented for two cases: $\alpha=1.1$ and $\alpha=1.4$. For each case, the five functions were implemented in a single program, where the parameters are fixed and the only code changed from one function to another was the evaluation function. Thus it appears that the Binary QPSO is extremely flexible and robust.

References

1. Angeline, P. J.: Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy and Performance Differences. Evolutionary Programming VIII, Lecture Notes in Computer Science 1447, Springer-Verlag (1998) 601-611
2. Clerc, M.: The Swarm and Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization. Proceedings of Congress on Evolutionary Computation, Piscataway, NJ (1999) 1951-1957
3. De Jong, K. A. "The analysis of the Behavior of a Class of Genetic Adaptive Systems. Ph. D. Dissertation, University of Michigan, Ann Arbor (1975)
4. Kennedy, J., Eberhart, R. C.: Particle Swarm Optimization. Proceedings of IEEE 1995 International Conference on Neural Network, IV. Piscataway, NJ (1995) 1942-1948
5. Kennedy, J.: Small Worlds and Mega-minds: Effects of Neighborhood Topology on Particle Swarm Performance. Proceedings of Congress on Evolutionary Computation, Piscataway, NJ (1999) 1931-1938
6. Kennedy, J., Eberhart, R. C.: A Discrete Binary Version of the Particles Swarm Algorithm. Proceedings of IEEE International Conference on Systems, Man and Cybernetics. Piscataway, NJ (1997) 4104-4108
7. Suganthan, P. N.: Particle Swarm Optimizer with Neighborhood Operator. Proceedings of 1999 Congress on Evolutionary Computation, Piscataway, NJ (1999) 1958-1962
8. Sun, J., Feng, B., Xu, W.-B.: Particle Swarm Optimization with Particles Having Quantum Behavior. Piscataway, NJ (2004) 325-331
9. Sun, J., Xu, W.-B., Feng, B.: A Global Search Strategy of Quantum-behaved Particle Swarm Optimization. Proceedings of IEEE 2004 Conference on Cybernetics and Intelligent Systems, Singapore (2004) 111-116
10. Shi, Y., Eberhart, R. C.: Empirical Study of Particle Swarm Optimization. Proceedings of 1999 Congress on Evolutionary Computation, Piscataway, NJ (1999) 1945-1950
11. Shi, Y., Eberhart, R.C.: A Modified Particle Swarm Optimizer. Proceedins of IEEE 1998 International Conference on Evolutionary Computation, Piscataway, NJ (1998) 1945-1950.

Artificial Environment for Simulation of Emergent Behaviour

Rafal Sienkiewicz and Wojciech Jedruch

Gdansk University of Technology, Gdansk, Poland
{Rafal.Sienkiewicz,Wojciech.Jedruch}@eti.pg.gda.pl

Abstract. The paper presents an artificial world model in which various self-organization and self-modification processes could be simulated. The model is a two dimensional space in which there are stacks of hexagonal tiles which are moving, colliding, and making bonds between them. On the higher level of organization a structure of tiles specifies some function whose execution affects other tiles in its neighborhood. The functions encoded in the structures of tiles are expressed in the simple Prolog like language. Few examples illustrate the behavior of the system.

1 Introduction

The aim of the work is to create some artificial, complete, low level, and closed environment consisting of space, moving objects, and rules which govern their interactions. Within the environment a complex behaviour of objects and emergent phenomena can be simulated. Especially a number of problems listed in [2] can be investigated e.g. simulations of spontaneous generation of life like systems, novel living organizations or open-ended evolution of life. The proposed system could be seen as an artificial life or an artificial chemistry model (see [13]).

The system operates on two levels. On the first level objects of the system move, collide, rebound and randomly change their structures. On the second level some structures of objects are capable of inducing changes in other objects. The types of changes are encoded in structures of objects in specially defined Prolog like language.

This work is a major modification of the model presented in papers [5,4] – a completely new language of objects and more realistic mechanics were developed. The presented systems is still developed and simulations are run. More about the system and actual projects can be found at the address www.swarm.eti.pg.gda.pl.

2 The Model

The environment is defined over two dimensional continuous space with periodic boundary condition.

2.1 Particles and Complexes

The constituent objects of the environment are called *particles* represented by hexagonal tiles. The particles are of 256 types and are characterized by velocity, position, and internal energy. Each type is related with a set of constant attributes: mass, bond energy (needed to disrupt bond between particles of given types), activation energy (needed to initiate any transformation of bonds).

Particles can bond together forming a *complex* of particles. The permanence of the complex depends on its constituent particles bond energies. Particles can bond vertically on the directions up and down forming stacks. The particles on the bottom of the stack can bond horizontally. An example of stack of particles and complex formed by horizontal bonds are shown in Fig. 1.

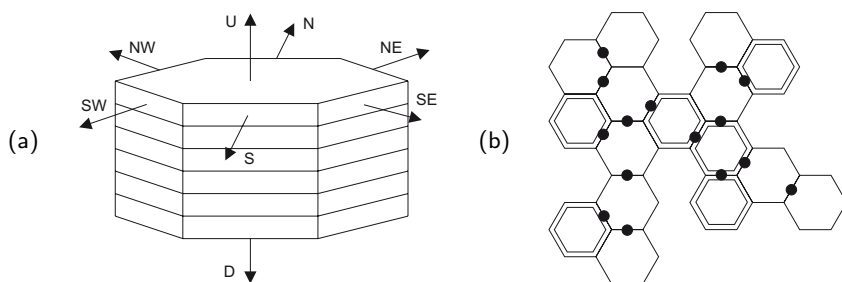


Fig. 1. Examples of complexes: (a) horizontal view of single stack of particles with directions shown, and (b) vertical view of complex formed by horizontal bonds, where hexagons drawn by single lines represent single particles, and by double lines represent stacks of particles and black dots mark horizontal bonds between particles

There are two types of collisions of particles and complexes: elastic and inelastic one, where the type is chosen randomly according to a preset probability. During elastic collision, the resulting velocities are calculated according to the rules of classical mechanics resolving the collision of two disks (circles surrounding hexagons) – conservation of momentum and of kinetic energy is observed. After inelastic collision the velocities of both particles are equalized – conservation of momentum is observed but the resulting decrease of kinetic energy of particles is compensated by emission of a *photon*.

2.2 Photons

In addition to permanent particles the environment contains temporary entities called *photons* which transport energy. They have no mass or momentum and move with constant velocity. They are characterized by energy, position and direction of movement.

Photons are created after inelastic collision of particles or complexes, after bonding of particles (bond energy is converted into a photon) or can be emitted spontaneously by particles – in each time step a particle may (with preset

probability) convert part of its internal energy into a new photon. Photons may collide with particles. There are two types of collisions: elastic and inelastic. Elastic collisions change photon direction only, after inelastic collision one of the following event is randomly selected (each event has its own preset probability):

1. Rebounding of the particle hit by the photon from an adjoining particle. The energy of the photon is transformed into the kinetic energy of rebounded particles.
2. Creating horizontal or vertical bond between the hit particle and adjoining particle. After the reaction a new photon is created having energy equal to the sum of: the energy of hitting photon, the energy of created bond and the decrease of kinetic energy of bonded particles.
3. Removing horizontal or vertical bond between the hit particle and bound particles. After the reaction a new photon is created having energy equal to the energy of hitting photon minus the energy of removed bond.
4. Photon absorption. Photon is absorbed by the particle, and is converted into its internal energy.

If selected event cannot be completed, e.g. because of insufficient photon energy, no other action is performed (photon skips the particle).

The above constitutes the first level of the system. When there are no photons and the probability of inelastic collisions of particles is set to zero the system behave like classical Newtonian one. Particles and complexes of particles move and collide. Otherwise, the possible inelastic collision of particles produce photons which in consequence can randomly change structures of other particles by creating and removing bonds between them.

2.3 Functions

Besides the reactions resulting from the collision of particles with photons, there exists additional class of interactions in which complexes of particles are capable to recognize and manipulate particular structures of particles in the space around them. The description of the function performed by a complex is contained in the types and locations of particles in the complex. The structure of the complex is then interpreted as a program written in a specially defined language described below.

2.4 Syntax

Syntax of the program encoded in a complex of particles is similar to the Prolog language, using the following predicates only: **program**, **search**, **action**, **structure**, **exists**, **bind**, **unbind**, **move**, **not**. Similarity to Prolog is clearly visible in the overall structure and execution algorithm but specific syntax of above predicates (especially **exists**) needs some transformation of the programs into the well-formed Prolog. Language syntax in BNF notation is shown in Table 1.

An example of a simple program is presented in Table 2. The program recognizes the structure shown in Fig. 2, and then binds the particle 11111111 to the unbound particle 00000000.

Table 1. Language syntax

```

program ::= program_header program_body
program_header ::= program() :- search(), action().
program_body ::= search action definitions
search ::= search() :- header .
action ::= action() :- row_action {, row_action } .
definitions ::= row_definition {row_definition }
row_definition ::= header :- body.
header ::= structure( integer )
body ::= exists( exists ) {,exists( exists ) } {,not( header ) }
      |not( header ) {,not( header ) }
short ::= 0|1|2|3|4|5|6|7|8|9
integer ::= short {short}
exists ::= [not] c [[not] bound [to f] [on d]] | [adjacent [to f] [on d]], [mark f]
row_action ::= bind( action_spec ) | unbind( unbind_spec ) | move( action_spec )
action_spec ::= f to f on d
unbind_spec ::= f [from f] [on d]
c ::= 0|1|×, 0|1|×, 0|1|×, 0|1|×, 0|1|×, 0|1|×, 0|1|×, 0|1|×
f ::= V short
d ::= N|NE|SE|S|SW|NW|U|D

```

Table 2. Example of a program recognizing the structure shown in Fig. 2

```

program():-
    search(), action().
search():-
    structure(0).
structure(0):-
    exists([0,0,0,0,0,0,×,×], mark V1),
    exists([1,1,1,1,1,1,1,1], bound to V1 on N, mark V2),
    exists([0,0,0,0,0,0,0,0], mark V5),
    not(structure(1)),
    not(structure(2)).
structure(1):-
    exists([1,1,1,1,0,0,0,0] bound to V2 on NW, mark V3),
    exists([1,1,1,1,0,0,0,0] bound to V3 on SW, mark V4),
    not(structure(3)).
structure(3):-
    exists([0,0,0,0,1,1,1,1] bound to V4 on S).
structure(2):-
    exists([1,0,1,0,1,0,1,0]).
action():-
    bind(V2 to V5 on SW).

```

2.5 Semantics

Program execution is based on Prolog backtracking algorithm. The predicates of the language may be divided into the following groups:

program, search, action, structure. Each program consists of the single, main predicate **program**. This predicate always calls two predicates: **search** and **action**. The first one calls searching predicates, while the second one calls execution predicates that can affect particles (by moving them or creating/removing bonds between them). The execution of "actions" are performed only if searching succeeded, i.e. particular structures were recognized. Searching commands are grouped in the **structure** predicates, which describes some particular structures. These predicates call both predicate **exists** and other predicates **structure**. Predicate **structure** embedded in another **structure** is always in its negative form, **not(structure())**, i.e. structure description is formed by the sequence of **exists** predicates and sequence of some negative condition.

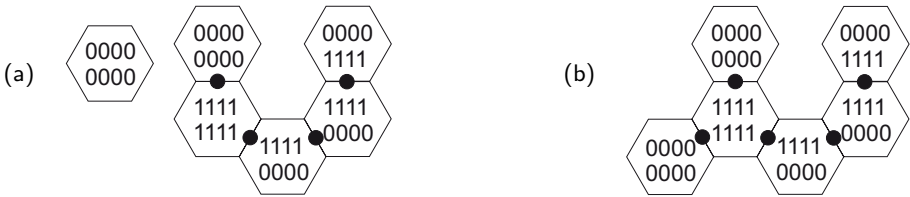


Fig. 2. Single particle and a complex of particles recognized by the program listed in Table 2 (a), and the structure after action of the program (b)

exists. The predicate **exists** is the basic command for structure searching. It recognizes a particle (or empty place adjoining a particle) of particular type, with particular bond structure.

bind, unbind, move. Predicates of the group affects particles in the system by moving them and/or making or removing bonds between them. If any of the predicate fails the whole program fails (backtracking is not performed).

not. Negates the single **structure** predicate.

2.6 Encoding

A complex of particles may encode a program. Each predicate **structure** is represented by the single stack of particles. Such a stack encodes a list of predicates **exists**. Stack which encodes **structure(0)** also encodes predicates **bind**, **unbind** and/or **move**. Adjoining stacks encode negative form of predicates **structure**. The program listed in Table 2 is represented by the structure of particles as shown in the Fig. 3.

Stacks encoding predicates **structure** are labelled by specific particles at the bottom of the stacks. Also inside the stacks, particles of specific types labels the beginning of predicates, eg. 00110000 encodes the beginning of predicate **exists**.

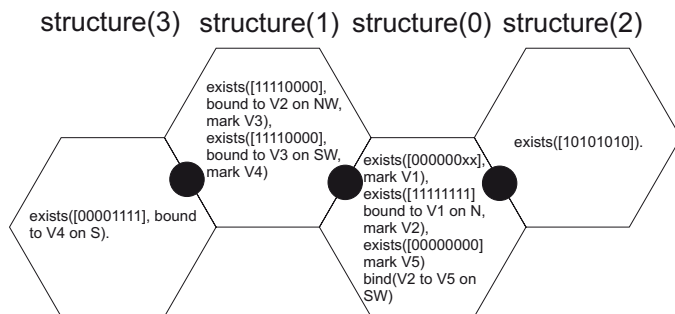


Fig. 3. Program listed in Table 2 encoded in complex of particles. Note that the predicates `not(structure)` are in stacks of particles which adjoin the `structure(0)` stack.

2.7 Interpreter

Programs encoded in complexes are translated into Prolog and run by built-in simplified interpreter. Just before execution, the interpreter creates list of facts about particles which are visible to program (particles which are contained in a circle of fixed radius surrounding the complex containing the program). Then the execution is performed. If both search and action part of the program succeeded the condition is checked that energy balance of all changes must be positive.

If all above conditions are fulfilled, the interpreter applies changes to environment. Otherwise, the program is rotated and executed again, i.e. before another execution, every direction related argument in predicates is changed according to rules: $N \rightarrow NE, NE \rightarrow SE, SE \rightarrow S, S \rightarrow SW, SW \rightarrow NW, NW \rightarrow N$. Only if all possibilities were tried and failed, the execution of program is cancelled.

2.8 Stages of Simulation

The simulation of the environment is running in epochs. In each epoch there are three phases. In the first one the movements and collisions of particles and complexes are realized. In the second phase the movement of photons and their collisions with particles are resolved. In the third phase the functions contained in complexes are executed in random order. During the execution of one function the rest of the system is frozen – only one function may be executed at a time.

3 Simulation Experiments

To illustrate the potential of the model a simulation experiment is presented. The aim of the experiment was to demonstrate how a set of complexes containing programs can construct a regular structure (“flake”) starting from a set of randomly distributed particles of two types.

The initial state of the environment consisted of 500 unbound particles of two types used as a building material and of six types of complexes containing

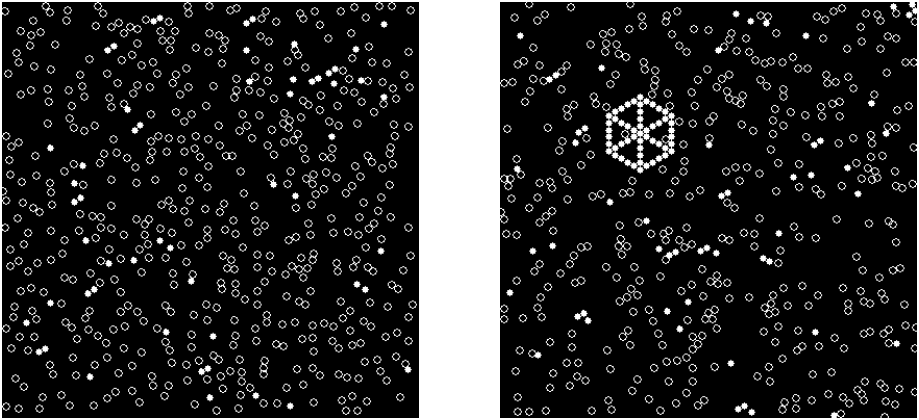


Fig. 4. Constructing the "flake". The initial state of the model, and the state after 1905 time steps. The white dots represents stacks of particles, in the left figure they contain programs.

programs. Each program existed in 8 copies (one of them in 6 copies). During the simulation the programs consequently built the structure. They worked in a sequence – the state of the structure left by one program was recognized by the second one and so on. The detailed description of the programs can be found in [7]. The initial state of the system and the state after 1905 time steps is shown in Fig. 4.

To check and demonstrate the "softness" of the language and the coding a few experiments were run in which some bugs were injected into the programs. This resulted in growing flakes of different shapes, but the erroneous set of programs still worked. So small changes in programs resulted in relatively small changes in its functioning. Two examples of erroneous behavior are shown in Fig. 5.

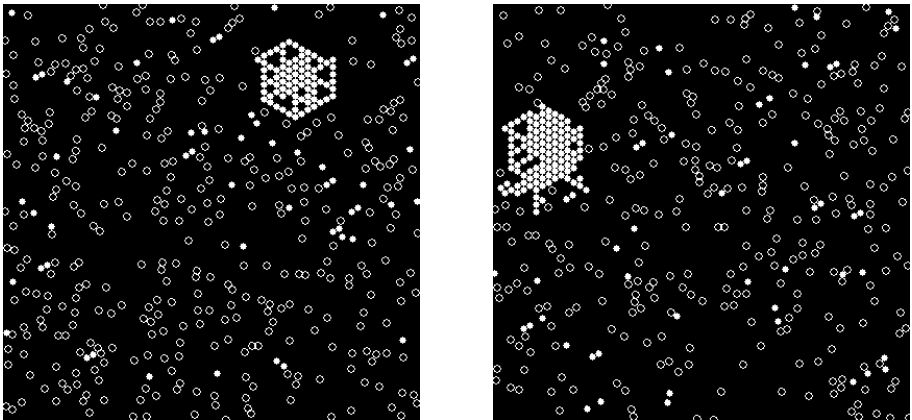


Fig. 5. Effects of bugs injected into the functions encoded in particles

4 Conclusions and Further Research

The artificial world model has been designed and implemented. Using the model a variety of selforganization and selfmodification processes can be modelled with no need to explicitly express a fitness of the units.

The implemented Prolog like language and the way of encoding it in the structure of complexes has the property that small changes in code of a program (i.e. in particles in which the program is encoded) usually lead to small changes in its execution effects. Such a property of the language is crucial while using the system to simulate the evolutionary, spontaneous development of complex structures. The lack of such a property was an obstacle in various simulations presented in [5,4,6].

The possible use of the model is planned in two directions. First the self replicating systems will be designed and modelled. Various self-replicating strategies can be compared – their dynamics and competition between them. By setting nonzero probabilities of reactions after the collisions with photons, the evolution of self-replicating systems can be investigated. The second direction is to model the evolution of system of particles starting from random initial states, looking for emergent properties of structures. The future research may extend the language to allow the recognition of the internal energy of distant particles which could give the possibility to develop some communication between distant structures.

References

1. Adami Ch., Introduction to artificial life, Springer, New York, (1998).
2. Bedau M.A., et al: Open Problems in Artificial Life, Artificial Life, v. 6, pp 363-376, (2000).
3. Dittrich P., Ziegler J., Banzhaf W.: Artificial Chemistries - A Review, Artificial Life, v. 7, pp. 225-275, (2001).
4. Jedruch W., Barski M., Experiments with a universe for molecular modelling of biological processes, BioSystems, v. 24, pp. 99-117, (1990).
5. Jedruch W., Sampson J., A universe for molecular modelling of self-replication, BioSystems v. 20, pp. 329-340, (1987).
6. Sienkiewicz R. and Jedruch W., Self-organization in Artificial Environment, Proc. VI Int. Conf. on AI, AI-19'2004, no. 23, pp. 81-88, Siedlce, Poland, (2004).
7. <http://www.swarm.eti.pg.gda.pl/>

A Novel and More Efficient Search Strategy of Quantum-Behaved Particle Swarm Optimization

Jun Sun¹, Choi H. Lai², Wenbo Xu¹, and Zhilei Chai¹

¹ Center of Intelligent and High Performance Computing,
School of Information Technology, Southern Yangtze University,
No. 1800, Lihudadao Road, Wuxi,
214122 Jiangsu, China

{sunjun_wx, xwb_sytu}@hotmail.com, zlchai@gmail.com

² School of Computing and Mathematical Sciences,
University of Greenwich, Greenwich, London SE10 9LS, UK
C.H.Lai@gre.ac.uk

Abstract. Based on the previous proposed Quantum-behaved Particle Swarm Optimization (QPSO), in this paper, a novel and more efficient search strategy with a selection operation is introduced into QPSO to improve the search ability of QPSO. While the center of position distribution of each particle in QPSO is determined by global best position and personal best position, in the Modified QPSO (MQPSO), the global best position is substituted by a personal best position of a randomly selected particle. The MQPSO also maintains the mean best position of the swarm as in the previous QPSO to make the swarm more efficient in global search. The experiment results on benchmark functions show that MQPSO has stronger global search ability than QPSO and PSO.

1 Introduction

Particle Swarm Optimization (PSO), originally proposed by J. Kennedy and R.C. Eberhart in 1995 [8], was introduced as a stochastic optimization method. PSO is considered as an evolutionary computation approach in that it possesses many characteristics that are used by evolutionary algorithms, such as initializing with a population of random solutions, searching for optima by updating generations, the adjustment of particles and evaluating particles by a fitness function. However unlike evolutionary algorithms, the updates of particles are not accomplished by selection, crossover or mutation, but implemented by simulation the social behavior of bird flock, fish school, insect swarm, animal herd and even human society.

In the past decade, PSO has been widely used in many real world applications and shown comparable performance with Genetic Algorithms (GAs). However, as demonstrated by F. Van Den Bergh [3], PSO is not a global convergence guaranteed algorithm because the particle is restricted to a finite sampling space for each of the iterations. This restriction weakens the global search ability of the algorithm and may lead to premature convergence in many cases. To overcome the shortcomings of the PSO, a Quantum-behaved Particle Swarm Optimization (QPSO) was proposed

previously [11], [12]. To improve QPSO further, in this paper, we introduce a selection operation into QPSO and thus propose a Modified QPSO.

The rest of the paper is structured as follows. In Section 2, the principle of the PSO is introduced. The concept of QPSO is presented in Section 3 and the Modified QPSO is proposed in Section 4. Section 5 gives the numerical results on some benchmark functions. Some concluding remarks and future work are presented in the last section.

2 Particle Swarm Optimization

In the original PSO with M individuals, each individual is treated as an infinitesimal particle in the D -dimensional space, with the position vector and velocity vector of particle i , $X_i(t) = (X_{i1}(t), X_{i2}(t), \dots, X_{iD}(t))$ and $V_i(t) = (V_{i1}(t), V_{i2}(t), \dots, V_{iD}(t))$. The particle moves according to the following equations:

$$V_{ij}(t+1) = V_{ij}(t) + c_1 \cdot r_1 \cdot (P_{ij}(t) - X_{ij}(t)) + c_2 \cdot r_2 \cdot (P_{gj}(t) - X_{ij}(t)) \tag{1}$$

$$X_{ij}(t+1) = X_{ij}(t) + V_{ij}(t+1) \tag{2}$$

for $i = 1, 2, \dots, M$; $j = 1, 2, \dots, D$. The parameters c_1 and c_2 are called the acceleration coefficients. Vector $P_i = (P_{i1}, P_{i2}, \dots, P_{iD})$ known as the *personal best position*, is the best previous position (the position giving the best fitness value so far) of particle i ; vector $P_g = (P_{g1}, P_{g2}, \dots, P_{gD})$ is the position of the best particle among all the particles and is known as the *global best position*. The parameters r_1 and r_2 are two random numbers distributed uniformly in $(0, 1)$, that is $r_1, r_2 \sim U(0, 1)$. Generally, the value of V_{ij} is restricted in the interval $[-V_{max}, V_{max}]$.

Many revised versions of PSO algorithm are proposed to improve the performance since its origin in 1995. Two most important improvements are the version with an Inertia Weight [15], w , and a Constriction Factor [4], K . In the inertia-weighted PSO the velocity is updated by using

$$V_{ij}(t+1) = w \cdot V_{ij}(t) + c_1 \cdot r_1 \cdot (P_{ij}(t) - X_{ij}(t)) + c_2 \cdot r_2 \cdot (P_{gj} - X_{ij}(t)) \tag{3}$$

while in the Constriction Factor model the velocity is calculated by using

$$V_{ij}(t+1) = K \cdot \left[V_{ij}(t) + c_1 \cdot r_1 \cdot (P_{ij}(t) - X_{ij}(t)) + c_2 \cdot r_2 \cdot (P_{gj} - X_{ij}(t)) \right] \tag{4}$$

where

$$K = \frac{2}{\left| 2 - \varphi - \sqrt{\varphi^2 - 4\varphi} \right|}, \quad \varphi = c_1 + c_2, \quad \varphi > 4 \tag{5}$$

The inertia-weighted PSO was introduced by Shi and Eberhart and is known as the Standard PSO [15]. The addition of inertia weight or Constriction Factor leads to faster convergence of the PSO algorithm. Other improvements of PSO can be seen in literatures such as [2], [9], [10], etc.

3 Quantum-Behaved Particle Swarm Optimization

Trajectory analyses in [5] demonstrated the fact that convergence of PSO algorithm may be achieved if each particle converges to its local attractor $p_i = (p_{1i}, p_{2i}, \dots, p_{id})$ with coordinates

$$p_{ij}(t) = (c_1 r_1 P_{ij}(t) + c_2 r_2 P_{gj}(t)) / (c_1 r_1 + c_2 r_2) \quad \text{or} \quad p_{ij}(t) = \varphi \cdot P_{ij}(t) + (1 - \varphi) \cdot P_{gj}(t) \quad (6)$$

where $\varphi = c_1 r_1 / (c_1 r_1 + c_2 r_2)$. It can be seen that the local attractor is a stochastic attractor of particle i that lies in a hyper-rectangle with P_i and P_g being two ends of its diagonal. We introduce the concepts of QPSO as follows.

Assume that each individual particle move in the search space with a δ potential on each dimension, of which the center is the point p_{ij} . For simplicity, we consider a particle in one-dimensional space, with point p the center of potential. Solving Schrödinger equation of one-dimensional δ potential well, we can get the probability distribution function $D(x) = e^{-2|p-x|/L}$. Using Monte Carlo method, we obtain

$$x = p \pm \frac{L}{2} \ln(1/u) \quad u \sim U(0,1) \quad (7)$$

The above is the fundamental iterative equation of QPSO.

In [12], a global point called Mainstream Thought or Mean Best Position of the population is introduced into PSO. The global point, denoted as C , is defined as the mean of the personal best positions among all particles. That is

$$C(t) = (C_1(t), C_2(t), \dots, C_D(t)) = \left(\frac{1}{M} \sum_{i=1}^M P_{i1}(t), \frac{1}{M} \sum_{i=1}^M P_{i2}(t), \dots, \frac{1}{M} \sum_{i=1}^M P_{iD}(t) \right) \quad (8)$$

where M is the population size and P_i is the personal best position of particle i . Then the value of L is evaluated by $L = 2\alpha |C_j(t) - X_{ij}(t)|$ and the position are updated by

$$X_{ij}(t+1) = p_{ij}(t) \pm \alpha \cdot |C_j(t) - X_{ij}(t)| \cdot \ln(1/u) \quad (9)$$

where parameter α is called Contraction-Expansion (CE) Coefficient, which can be tuned to control the convergence speed of the algorithms. Generally, we always call the PSO with equation (9) Quantum-behaved Particle Swarm Optimization (QPSO), where parameter α must be set as $\alpha < 1.782$ to guarantee convergence of the particle [13]. In most cases, α can be controlled to decrease linearly from α_0 to α_1 ($\alpha_0 < \alpha_1$).

4 The Proposed Algorithm

QPSO is a promising optimization problem solver that outperforms PSO in many real application areas. First of all, the introduced exponential distribution of positions makes QPSO global convergent. Furthermore, the introduction of mean best position

into QPSO is another improvement of QPSO. In original PSO, each particle converges to the global best position independently. On the other hand, in the QPSO with mean best position C , each particle cannot converge to global best position without regard to its colleagues for there are wait among the particles as Fig.1 shows. It is because that the distance between particle's current position and C determines the position distribution of the particle for next iteration. If the personal best positions of several particles are far from the global best position (these particle called lagged particles) while those of the other particles are near the P_g , the position C may be pulled away from P_g , by lagged particles. When the lagged particles are chasing after their colleagues, say converging to P_g , the position C will be approaching P_g , slowly. The distances between position C and the personal best positions of a particle near P_g , don't decrease quickly, decelerating the convergences of the particles near P_g , and making them explore globally around P_g , temporarily until the lagged ones are close to P_g . Therefore, in the QPSO with mean best position, the particle swarm does never abandon any lagged particle and seems to be more intelligent and more cooperative social organism.

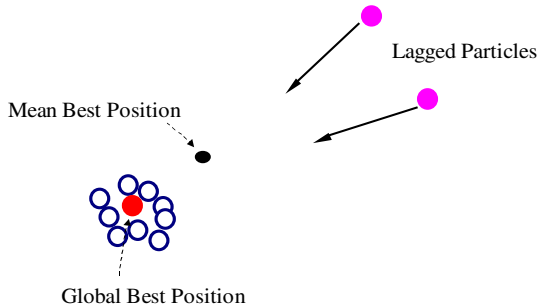


Fig. 1. The example shows there are wait among particles in QPSO

Like original PSO, all particles in QPSO are converging to the global best position P_g , on the course of the search, which make the convergence speed of algorithms relatively faster than Genetic Algorithms (GAs) and therefore QPSO or PSO could generate the solution with higher precision. However, the memories of particles and their convergence to P_g , maybe lead to premature convergence. Let's take Fig.1 as an example. In Fig. 1, most of the particles are clustering around the global best position, toward which the lagged particles will be pulled. In case that the global optima lies in the area where the lagged particles locate, the optimal solutions may be escaped by the particles since P_g is so far from the global optima that the particles around P_g , could have little opportunity to hit it. The lagged particles may also miss the global optima because they are rushing at P_g , and appear in the promising area with little probability. Unless the particles are lucky enough to search out it during the process of converging, the probability with which the global optima may be found is decreasing, leading the algorithm to premature convergence as a result.

To overcome this shortcoming, we introduce selection operation into QPSO. The selection operation, not like what proposed by Angeline [2], is exerted on the global best solution. That is, at each iteration, the particle's local attract point p_i is not determined by global best position, but by a certain position G that is randomly selected from the other particle's personal best positions and has better fitness value than the particle. The selection operation can be described as follows.

Selection_G:

```

randomly select a particle k from the swarm;
if  $f(P_k) < f(P_i)$ 
     $G = P_k$ ;
else
     $G = P_g$ ;
endif
Return  $G$ 

```

Accordingly, the coordinate of p_i is calculated by

$$p_{ij}(t) = (c_1 r_1 P_{ij}(t) + c_2 r_2 G_j(t)) / (c_1 r_1 + c_2 r_2), \text{ or } p_{ij}(t) = \varphi \cdot P_{ij}(t) + (1 - \varphi) \cdot G_j(t) \quad (10)$$

Thus each particle's personal best position may be selected as position G and be a point that the particles converge to. For example, in Fig.1, a lagged particle may be selected as the G position, and if so, other particles will run to it and may find the global optima in the promising area easily. Therefore, it can be concluded that this selection mechanism could enhance considerably the opportunity of particles' finding the global optima.

The proposed Modified QPSO outlined as follows.

Modified QPSO (MQPSO) Algorithm

```

Initialize particles with random position  $X_i = X[i][:]$ ;
Initialize personal best position by set  $P_i = X_i$ ;
while the stop criterion is not met do
    Compute the mean best position  $C[:]$  by equation (8);
    for  $i = 1$  to swarm size  $M$ 
        if  $f(X_i) < f(P_i)$  then  $P_i = X_i$ ; endif
        Find the  $P_g = \arg \min f(P[g][:])$ ;
         $G = \text{Selection\_G}$ ;
        for  $j = 1$  to  $D$ 
             $\varphi = \text{rand}(0, 1)$ ;  $u = \text{rand}(0, 1)$ ;
             $p = \varphi * P[i][j] + (1 - \varphi) * G[j]$ ;
            if ( $\text{rand}(0, 1) > 0.5$ )
                 $X[i][j] = p + \alpha * \text{abs}(C[j] - X[i][j]) * \ln(1/u)$ ;
            else
                 $X[i][j] = p - \alpha * \text{abs}(C[j] - X[i][j]) * \ln(1/u)$ ;
            endif
        endfor
    endfor
endwhile

```

The values of parameter α can be controlled as in the QPSO [11], [12], [13]. In our experiment for this paper, α varies linearly over the running of algorithm.

5 Experiment

Five widely known benchmark functions listed in Table 1 are tested for the performance comparison of the Modified QPSO (MQPSO) with Standard PSO (SPSO) and QPSO algorithms. These functions are all minimization problems with minimum objective function values zeros. The initial range of the population listed in Table 2 is asymmetry as used in [14], [15]. Table 2 also lists V_{max} for SPSO.

Table 1. Expression of the five tested benchmark functions

| | Function Expression | Search Domain |
|------------|--|--------------------------|
| Sphere | $f_1(X) = \sum_{i=1}^n x_i^2$ | $-100 \leq x_i \leq 100$ |
| Rosenbrock | $f_2(X) = \sum_{i=1}^{n-1} (100 \cdot (x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$ | $-100 \leq x_i \leq 100$ |
| Rastrigrin | $f_3(X) = \sum_{i=1}^n (x_i^2 - 10 \cdot \cos(2\pi x_i) - 10)$ | $-10 \leq x_i \leq 10$ |
| Greiwank | $f_4(X) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ | $-600 \leq x_i \leq 600$ |
| Shaffer's | $f_5(X) = 0.5 + \frac{(\sin(\sqrt{x_1^2 + x_2^2}))^2}{(1.0 + 0.001(x_1^2 + x_2^2))^2}$ | $-100 \leq x_i \leq 100$ |

The fitness value is set as function value and the neighborhood of a particle is the whole population. We had 50 trial runs for every instance and recorded mean best fitness and standard deviation. In order to investigate the scalability of the algorithm, different population sizes M are used for each function with different dimensions. The population sizes are 20, 40 and 80. The maximum generation (iteration) is set as 1000, 1500 and 2000 corresponding to the dimensions 10, 20 and 30 for first four functions, respectively. The maximum generation for the last function is 2000. For SPSO, the acceleration coefficients are set to be $c_1=c_2=2$ and the inertia weight is decreasing linearly from 0.9 to 0.4 as in [14], [15]. In experiments for QPSO, the value of CE Coefficient α varies from 1.0 to 0.5 linearly over the running of the algorithm as in [12], [13], while in MQPSO, the value of α decreases from 1.0 to 0.4 linearly. The mean values and standard deviations of best fitness values for 50 runs of each function are recorded in Table 3 to Table 7.

The numerical results show that both QPSO and MQPSO are superior to SPSO except on Shaffer's f6 function. On Shpere Function the QPSO works better than

Table 2. The initial range of population for all the tested algorithms and V_{max} for SPSO

| | Initial Range | V_{max} |
|-------|---------------|-----------|
| f_1 | (50, 100) | 100 |
| f_2 | (15, 30) | 100 |
| f_3 | (2.56, 5.12) | 10 |
| f_4 | (300, 600) | 600 |
| f_5 | (30, 100) | 100 |

Table 3. Numerical results on Sphere function

| M | Dim. | Gmax | SPSO | | QPSO | | MQPSO | |
|----|------|------|-----------|----------|-----------|----------|-----------|-----------|
| | | | Mean Best | St. Dev. | Mean Best | St. Dev. | Mean Best | St. Dev. |
| 20 | 10 | 1000 | 3.16E-20 | 6.23E-20 | 2.29E-41 | 1.49E-40 | 1.55E-038 | 9.96E-038 |
| | 20 | 1500 | 5.29E-11 | 1.56E-10 | 1.68E-20 | 7.99E-20 | 6.16E-021 | 2.26E-020 |
| | 30 | 2000 | 2.45E-06 | 7.72E-06 | 1.34E-13 | 3.32E-13 | 4.86E-014 | 1.34E-013 |
| 40 | 10 | 1000 | 3.12E-23 | 8.01E-23 | 8.26E-72 | 5.83E-71 | 3.93E-060 | 2.74E-059 |
| | 20 | 1500 | 4.16E-14 | 9.73E-14 | 1.53E-41 | 7.48E-41 | 8.95E-038 | 1.89E-037 |
| | 30 | 2000 | 2.26E-10 | 5.10E-10 | 1.87E-28 | 6.73E-28 | 5.37E-026 | 2.36E-025 |
| 80 | 10 | 1000 | 6.15E-28 | 2.63E-27 | 3.10E-100 | 2.10E-99 | 7.57E-082 | 5.31E-081 |
| | 20 | 1500 | 2.68E-17 | 5.24E-17 | 1.56E-67 | 9.24E-67 | 9.08E-053 | 5.13E-052 |
| | 30 | 2000 | 2.47E-12 | 7.16E-12 | 1.10E-48 | 2.67E-48 | 2.77E-040 | 7.29E-040 |

Table 4. Numerical results on Rosenbrock function

| M | Dim. | Gmax | SPSO | | QPSO | | MQPSO | |
|----|------|------|-----------|----------|-----------|----------|-----------|----------|
| | | | Mean Best | St. Dev. | Mean Best | St. Dev. | Mean Best | St. Dev. |
| 20 | 10 | 1000 | 94.1276 | 194.3648 | 59.4764 | 153.0842 | 21.2251 | 49.1371 |
| | 20 | 1500 | 204.337 | 293.4544 | 110.664 | 149.5483 | 81.7652 | 118.7919 |
| | 30 | 2000 | 313.734 | 547.2635 | 147.609 | 210.3262 | 114.0708 | 160.1809 |
| 40 | 10 | 1000 | 71.0239 | 174.1108 | 10.4238 | 14.4799 | 6.7391 | 8.9291 |
| | 20 | 1500 | 179.291 | 377.4305 | 46.5957 | 39.5363 | 41.1235 | 41.6602 |
| | 30 | 2000 | 289.593 | 478.6273 | 59.0291 | 63.4941 | 47.8311 | 32.7463 |
| 80 | 10 | 1000 | 37.3747 | 57.4734 | 8.63638 | 16.6746 | 7.0730 | 8.2491 |
| | 20 | 1500 | 83.6931 | 137.2637 | 35.8947 | 36.4702 | 28.9102 | 28.6860 |
| | 30 | 2000 | 202.672 | 289.9728 | 51.5479 | 40.8492 | 55.2497 | 41.1081 |

MQPSO and SPSO. It is because that selection operation in MQPSO may enhance global search ability but weaken the local search ability of QPSO at later search stage. On Rosenbrock function, the MQPSO outperforms the QPSO except when the swarm size is 80 and dimension is 30. On Rastrigrin function, it is also shown that the MQPSO generated better results in most cases. On Griewank function, MQPSO is superior to the QPSO when the dimension of the problem is 20 and 30. When

dimension is 10, MQPSO cannot generate better solution than QPSO under this parameter setting. On Shaffer’s function, the MQPSO shows its stronger ability to escape the local minima 0.0097 than the QPSO and even than SPSO. Generally speaking, the MQPSO has better global search ability than QPSO.

Fig. 2 shows the convergence process of the MQPSO and QPSO on the first four benchmark functions with dimension 30 and swarm size 20. It is shown that, although MQPSO converge more slowly than the QPSO during the early stage of search, it may catch up with QPSO at later stage and could generate better solutions at the end of search.

Table 5. Numerical results on Rastrigrin function

| M | Dim. | Gmax | SPSO | | QPSO | | MQPSO | |
|----|------|------|-----------|----------|-----------|----------|-----------|----------|
| | | | Mean Best | St. Dev. | Mean Best | St. Dev. | Mean Best | St. Dev. |
| 20 | 10 | 1000 | 5.5382 | 3.0477 | 5.2543 | 2.8952 | 3.9190 | 3.0742 |
| | 20 | 1500 | 23.1544 | 10.4739 | 16.2673 | 5.9771 | 16.9898 | 13.0674 |
| | 30 | 2000 | 47.4168 | 17.1595 | 31.4576 | 7.6882 | 24.3047 | 7.4604 |
| 40 | 10 | 1000 | 3.5778 | 2.1384 | 3.5685 | 2.0678 | 2.6979 | 2.0492 |
| | 20 | 1500 | 16.4337 | 5.4811 | 11.1351 | 3.6046 | 10.3782 | 9.2972 |
| | 30 | 2000 | 37.2796 | 14.2838 | 22.9594 | 7.2455 | 17.7308 | 4.0300 |
| 80 | 10 | 1000 | 2.5646 | 1.5728 | 2.1245 | 2.2353 | 2.2060 | 2.3589 |
| | 20 | 1500 | 13.3826 | 8.5137 | 10.2759 | 6.6244 | 7.7723 | 5.9316 |
| | 30 | 2000 | 28.6293 | 10.3431 | 16.7768 | 4.4858 | 13.1504 | 3.2067 |

Table 6. Numerical results on Griewank function

| M | Dim. | Gmax | SPSO | | QPSO | | MQPSO | |
|----|------|------|-----------|----------|-----------|----------|-----------|----------|
| | | | Mean Best | St. Dev. | Mean Best | St. Dev. | Mean Best | St. Dev. |
| 20 | 10 | 1000 | 0.09217 | 0.0833 | 0.08331 | 0.06805 | 0.0833 | 0.0721 |
| | 20 | 1500 | 0.03002 | 0.03255 | 0.02033 | 0.02257 | 0.0162 | 0.0171 |
| | 30 | 2000 | 0.01811 | 0.02477 | 0.01119 | 0.01462 | 0.0063 | 0.0088 |
| 40 | 10 | 1000 | 0.08496 | 0.0726 | 0.06912 | 0.05093 | 0.0745 | 0.0724 |
| | 20 | 1500 | 0.02719 | 0.02517 | 0.01666 | 0.01755 | 0.0122 | 0.0122 |
| | 30 | 2000 | 0.01267 | 0.01479 | 0.01161 | 0.01246 | 0.0061 | 0.0110 |
| 80 | 10 | 1000 | 0.07484 | 0.07107 | 0.03508 | 0.02086 | 0.0499 | 0.0642 |
| | 20 | 1500 | 0.02854 | 0.0268 | 0.0146 | 0.01279 | 0.0077 | 0.0115 |
| | 30 | 2000 | 0.01258 | 0.01396 | 0.01136 | 0.01139 | 0.0063 | 0.0104 |

Table 7. Numerical results on Shaffer’s f6 function

| M | Dim. | Gmax | SPSO | | QPSO | | MQPSO | |
|----|------|------|-----------|-----------|-----------|-----------|------------|------------|
| | | | Mean Best | St. Dev. | Mean Best | St. Dev. | Mean Best | St. Dev. |
| 20 | 2 | 2000 | 2.782E-04 | 0.001284 | 0.001361 | 0.003405 | 1.945E-004 | 0.0014 |
| 40 | 2 | 2000 | 4.744E-05 | 3.593E-05 | 3.891E-04 | 0.001923 | 3.256E-008 | 1.087E-007 |
| 80 | 2 | 2000 | 2.568E-10 | 3.134E-10 | 1.723E-09 | 3.303E-09 | 6.517E-009 | 1.428E-008 |

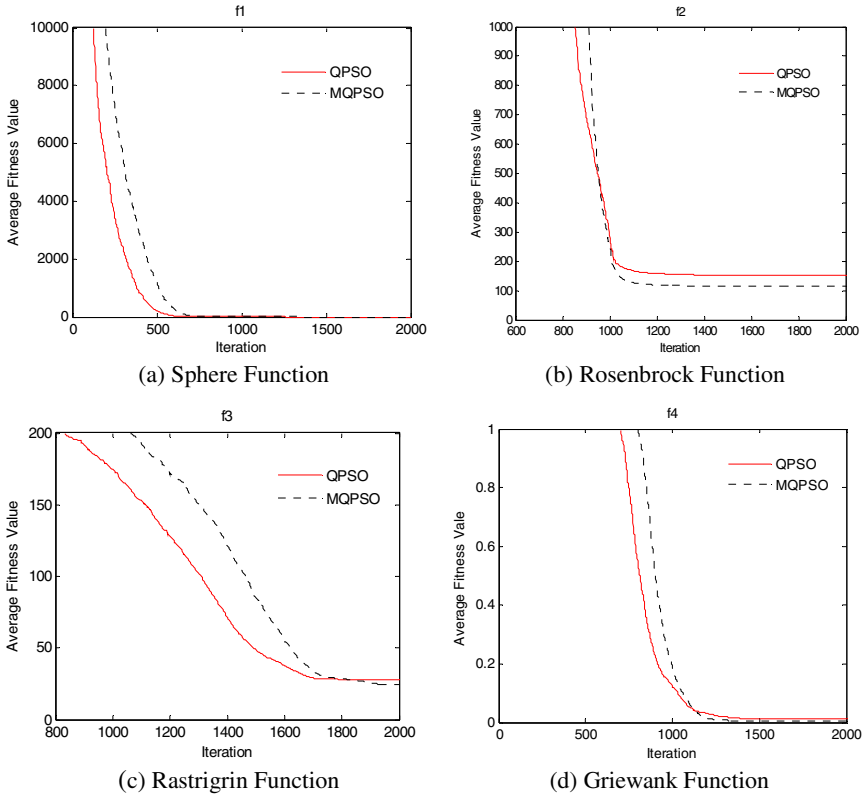


Fig. 2. Convergence process of the MQPSO and QPSO on the first four benchmark functions with dimension 30 and swarm size 20 averaged on 50 trail runs

6 Conclusion

In this paper, a Modified QPSO with a selection operation is proposed. The motive of introducing the selection into QPSO is that particles in QPSO or original PSO converge to the global best position, which may be apt to encounter premature convergence. The selection operation exerted on global best position is able to enhance the global search ability of QPSO considerably as shown by the performance comparison between MQPSO and QPSO.

Our future work for QPSO will focus on introducing into QPSO more selection operation method and even other evolutionary operations. Moreover, we will also be devoted to applying the MQPSO and QPSO to some real world problems.

References

1. Angeline, P.J.: Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy and performance Differences. Evolutionary Programming VII, Lecture Notes in Computer Science 1447. Springer-Verlag, Heidelberg (1998) 601-610

2. Angeline, P.J.: Using Selection to Improve Particle Swarm Optimization. Proc. 1998 IEEE International Conference on Evolutionary Computation. Piscataway, NJ (1998) 84-89
3. Van den Bergh, F.: An Analysis of Particle Swarm Optimizers. PhD Thesis. University of Pretoria, South Africa (2001)
4. Clerc, M.: The Swarm and Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization. Proc. 1999 Congress on Evolutionary Computation. Piscataway, NJ (1999) 1951-1957
5. Clerc, M., Kennedy, J.: The Particle Swarm: Explosion, Stability, and Convergence in a Multi-dimensional Complex Space. IEEE Transactions on Evolutionary Computation, Vol. 6, No. 1. Piscataway, NJ (2002) 58-73
6. Eberhart, R.C., Shi, Y.: Comparison between Genetic Algorithm and Particle Swarm Optimization. Evolutionary Programming VII, Lecture Notes in Computer Science 1447, Springer-Verlag, Heidelberg (1998) 611-616
7. Holland, J.H.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, Michigan (1975)
8. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. Proc. IEEE 1995 International Conference on Neural Networks, IV. Piscataway, NJ (1995) 1942-1948
9. Kennedy, J.: Small worlds and Mega-minds: Effects of Neighborhood Topology on Particle Swarm Performance. Proc. 1999 Congress on Evolutionary Computation. Piscataway, NJ (1999) 1931-1938
10. Suganthan, P.N.: Particle Swarm Optimizer with Neighborhood Operator. Proc. 1999 Congress on Evolutionary Computation, Piscataway, NJ (1999) 1958-1962
11. Sun, J., Feng, B., Xu, W.-B.: Particle Swarm Optimization with Particles Having Quantum Behavior. Proc. 2004 Congress on Evolutionary Computation, Piscataway, NJ (2004) 325-331
12. Sun, J., Xu, W.-B., Feng, B.: A Global Search Strategy of Quantum-behaved Particle Swarm Optimization. Proc. 2004 IEEE Conference on Cybernetics and Intelligent Systems, Singapore (2004) 111-115
13. Sun, J., Xu, W.-B., Feng, B.: Adaptive Parameter Control for Quantum-behaved Particle Swarm Optimization on Individual Level. Proc. 2005 IEEE International Conference on Systems, Man and Cybernetics. Piscataway, NJ (2005) 3049-3054
14. Shi, Y., Eberhart, R.: Empirical Study of Particle Swarm Optimization. Proc. 1999 Congress on Evolutionary Computation. Piscataway, NJ (1999) 1945-1950
15. Shi, Y., Eberhart, R.C.: A Modified Particle Swarm. Proc. 1998 IEEE International Conference on Evolutionary Computation. Piscataway, NJ (1998) 69-73

Extracting Grammars from RNA Sequences

Gabriela Andrejková, Helena Lengeňová, and Michal Mati*

Institute of Computer Science, Faculty of Science

P. J. Šafárik University, Košice, Slovakia

{Gabriela.Andrejkova,Michal.Mati}@upjs.sk, Hela.Lengenova@gmail.com

Abstract. In the paper, we describe an application of stochastic context-free grammars (SCFG) to modelling of the formal RNA string language. The simplification of the stochastic context-free grammar and its conversion to Chomsky normal form was used. We present the modification of Cocke-Kasami-Younger algorithm that is used for probabilistic estimations of stochastic grammars for RNA sequences. Some better algorithms were constructed to decrease the computational complexity but still on the level of $O(n^3)$ where n is the length of the RNA strings. The results of using the algorithms to the training sample consisted of tRNA chains of *Acinetobacter* sp. bacteria are described.

1 Introduction

Algorithms for an analysis of biological sequences, RNA, DNA and proteins molecules as strings of nucleotide or amino acid are still interesting problems. Most algorithms for the analysis assume *uncorrelated* strings of residues, in which the identity of a residue at one position has no effect on the identity of another residue (Rivas and Eddy [8]). It means, the analysis of such sequences is very reduced, because RNA secondary structure produces strong long-distance pairwise correlations between Watson-Crick base pairs.

Computational linguistics gives us another possibility to model strings using *correlated* symbols (Jurafsky and Martin [5]), (Gold [3]). From the point of view, a language of RNA is dominated by *nested* pairwise correlations which should be constructed using a context-free grammar. Stochastic context-free grammars (SCFG) have been used to create probabilistic model that describes RNA secondary structure (Sakakibara and Muramatsu [10], [11]). In the case, it is necessary to apply some algorithms that learn from RNA strings as examples of a concept space. Two aspects must be considered with regards to the learning of SCFGs: (1) the learning of the structural component, i. e. the rules of grammars, (2) the estimation of the stochastic component, i. e. the probabilities of the rules. Classical inductive techniques that are based on grammatical inference to obtain SCFGs have been proposed. However, these techniques present computational restrictions that limit their use in complex real tasks (Linares et al. [6]). Genetic algorithms have been applied to the grammar inference problem from programs (Črepinšek et al. [2]). It was proved that the property of consistency is guaranteed

* Supported by the Slovak Scientific Grant Agency VEGA, Grant No. 1/3128/06.

for all the SCFGs constructed by using the classical Inside-Outside and Viterbi algorithms (Sanchez and Benedi [9]).

In the paper, in the following section we describe a grammar construction from RNA strings, we analyse probabilistic estimations of rules in stochastic grammars and its consistency. In the third section we show the construction of the stochastic grammar in Chomsky normal form from a common form and we recomputed probabilities of rules from the theoretical point of view. In the fourth section Cocke-Kasami-Younger algorithm is modified to a learning of grammars from RNA strings. We developed some modifications of the algorithm to get better time complexity. The next section contains results of the approach in an application of the method to tRNA of *Acinetobacter* sp. bacteria. In the conclusion we summarize the results.

2 Construction of the Grammar

We consider context-free grammars (CFG) as sufficient to at least partially cover the most important properties of RNA strings while maintaining polynomial time complexity of computations based on them. In the first step, we can model a simple grammar G consisting of one non-terminal X (which is also the starting symbol), four terminals - a, g, c, u - corresponding to four nucleotide bases and four sets of the following types of rules:

- (1) $X \rightarrow a|g|c|u,$
- (2) $X \rightarrow aX|gX|cX|uX,$ (3) $X \rightarrow Xa|Xg|Xc|Xu$
- (4) $X \rightarrow aXu|cXg|gXc|uXa$ (complementarity)

The stochastic grammar (SG) G_S can be easily obtained by assigning a probability vector Π . This grammar is able to cope with the following patterns of secondary RNA structure: stems, loops and bulges. This approach suffers from a great shortcoming. Although it is very simple, there are 2^n possible derivations for each word of the size n . Since G is CFG we can consider possible improvement by transformation to some normal form (Greibach or Chomsky). This transformation, however, is not so easy when we work with the stochastic grammar. A probability of each new or changed rule must be recomputed using probabilistic rules and methods [10], [4].

Notation: $P_G(X \rightarrow \alpha)$ is a probability of $X \rightarrow a$ rule in G_S grammar, $P_G(\alpha)$ is a probability that α is derived from the starting symbol in G_S . $P_G(\beta \rightarrow \alpha)$ is a probability of deriving α from β in G_S .

We say that stochastic grammars $G_{S_1} = (G_1, \Pi_1)$ and $G_{S_2} = (G_2, \Pi_2)$ generate the same language iff $P_{G_1}(\alpha) = P_{G_2}(\alpha)$ for each string α .

Let G_s be SG. $R_{G_s}(X) = \{(\alpha_1, p_1), \dots, (\alpha_m, p_m)\}$ is a set of tuples representing all strings and respective probabilities such that G_S contains $X \rightarrow \alpha_i$ rule with probability $p_i, i \in [1, m]$. Then we can use the following properties (they can be proved) to the construction of the SCFG.

1. We can omit all the rules with zero probability.
2. If G_S contains a non-terminal X different from the starting symbol, which is on the left-hand-side of some rules, but not on the right-hand side. Then G_S without rules containing X generate the same language.
3. If X is a non-terminal in G_s and there exists a non-terminal $Y \in G_s, Y \neq X$, such that $R_{G_s}(Y) = R_{G_s}(X)$ then it is possible to replace every occurrence of Y on the right-hand-side of the rules contained in G_S by X without any influence on the language generated by G_S .

We will consider using of some learning algorithm to a stochastic grammar and we should answer a question if it is possible to simplify the grammar using the previously mentioned methods without changing the zero-probability rules to non-zero (during the learning process) or not.

3 Chomsky Normal Forms

For the future goals we analysed process of the modification of SCFG to the Chomsky normal form (CNF). We present our results in short and we focused on (1) the construction and (2) the consistency problem.

Theorem 1. *Let G be a CFG without any rule with empty symbol on the right-hand-side and $G_S = (G, \Pi)$ corresponding stochastic grammar. Let G' be a CFG in Chomsky normal form such that $L(G') = L(G) - \epsilon$. Then a probability vector Π' such that $G'_S = (G', \Pi')$ generates the same language as G_S exists.*

The theorem enables to transform the grammar G_S to $G_S^1 = (G^1, \Pi^1)$ such that G_S^1 is in CNF and generates the same language as G_S . The rules of G_S^1 are:

1. Nor it is necessary to modify the rules of the first type neither the corresponding probabilities: $X \rightarrow a|c|g|u$,
2. The rules of the second type are replaced by new rules:
 $X \rightarrow X_a^2 X | X_c^2 X | X_g^2 X | X_u^2 X$, the rules have the same probabilities as the rules in G_S ,
 $X_a^2 \rightarrow a, X_c^2 \rightarrow c, X_g^2 \rightarrow g, X_u^2 \rightarrow u$, the probabilities are 1.
3. The third type rules are transformed to:
 $X \rightarrow X X_a^3 | X X_c^3 | X X_g^3 | X X_u^3$, probabilities are the same as in the original rules' from G_S ,
 $X_a^3 \rightarrow a, X_c^3 \rightarrow c, X_g^3 \rightarrow g, X_u^3 \rightarrow u$, the probabilities are equal 1,
4. Instead of the rules of the fourth type G_S^1 contains:
 $X \rightarrow X_a^4 X_{X_u} | X_c^4 X_{X_g} | X_g^4 X_{X_c} | X_u^4 X_{X_a}$, the probabilities are unchanged,
 $X_{X_a} \rightarrow X X_a^5, X_{X_c} \rightarrow X X_c^5, X_{X_g} \rightarrow X X_g^5, X_{X_u} \rightarrow X X_u^5$, the probabilities are 1,
 $X_a^4 \rightarrow a, X_c^4 \rightarrow c, X_g^4 \rightarrow g, X_u^4 \rightarrow u$, the probabilities are 1,
 $X_a^5 \rightarrow a, X_c^5 \rightarrow c, X_g^5 \rightarrow g, X_u^5 \rightarrow u$, again the probabilities are 1.

Each nonterminal $X_a^i, i = 2, \dots, 5, X_a^i \in G_S^1$ rewrites one rule, furthermore right-hand side of the rules is unaltered hence we can substitute the nonterminals

$X_a^i, i = 3, 4, 5$ on the right-hand side by a nonterminal X_a^2 . Then $X_a^i, i = 3, 4, 5$ occur only on the left-hand side of the rules, therefore we can omit the rules rewriting them. The same can be done with nonterminals $X_b^i, i = 2, \dots, 5, b \in \{c, g, u\}$. G_S^1 can therefore be simplified to the grammar G_S^2 as follows:

$$\begin{aligned}
 X &\rightarrow a|c|g|u, X_a^2 \rightarrow a, X_c^2 \rightarrow c, X_g^2 \rightarrow g, X_u^2 \rightarrow u, X \rightarrow X_a^2 X | X_c^2 X | X_g^2 X | X_u^2 X, \\
 X &\rightarrow X X_a^2 | X X_c^2 | X X_g^2 | X X_u^2, X \rightarrow X_a^2 X_{Xu} | X_c^2 X_{Xg} | X_g^2 X_{Xc} | X_u^2 X_{Xa}, \\
 X_{Xa} &\rightarrow X X_a^2, X_{Xc} \rightarrow X X_c^2, X_{Xg} \rightarrow X X_g^2, X_{Xu} \rightarrow X X_u^2.
 \end{aligned}$$

The number of possible derivations of words after the above mentioned transformations is intact as each rule used in derivation of some word in G_S is only substituted by a sequence of rules determined by normalization process in G_S^2 . An estimation of the number of possible derivations of a word $w, |w| = n$ in CNF grammar: Let $N = \{A_1, \dots, A_r\}$ be a set of its non-terminals and M_i the number of rules, that rewrite A_i to two non-terminals. We denote the count of all possible derivations of a word of length n by $C(n)$. Then for each $n \geq 2$: $C(n) \leq k * (k + 1)^{(2n-4)}$, where $k = \max_{\{i \in [1, r]\}} M_i$.

4 Probabilistic Estimations of Rules in SCFG

The Cocke-Kasami-Younger (CYK) algorithm [5] uses the dynamic programming method to determine if some string belongs to a language generated by some CFG in Chomsky normal form. The time complexity of CYK algorithm is $O(n^3)$ and we describe the CYK algorithms for some better understanding of it.

4.1 Cocke-Kasami-Younger Algorithm

Input: string α , a sequence of symbols a_1, \dots, a_n . Let G contains r non-terminals: S_1, \dots, S_r . Let S_1 be a starting symbol. $P[i, j, k]$ is a three-dimensional Boolean array; $i, j \in [1, n]; k \in [1, r]$. $P[i, j, k]$ is true iff we can derive a subsequence of α starting with a_i of length j from a non-terminal S_k .

1. **assign** FALSE to all elements of P;
2. **for** each $i \in [1, n]$ **for** each $S_k \rightarrow a_i$ rule **set** $P[i, 1, k]$ to TRUE;
3. **for** each $i \in [2, n]$ **for** each $j \in [1, n - i + 1]$ **for** each $k \in [1, i - 1]$ **for** each $S_A \rightarrow S_B S_C$ rule **if** $P[j, k, B]$ and $P[j + k, i - k, C]$ are both TRUE **then set** $P[j, i, A]$ to TRUE.
4. **if** $P[1, n, 1]$ is TRUE, **then** $\alpha \in L(G)$, **otherwise** $\neg(\alpha \in L(G))$.

4.2 Extending CYK for Learning Algorithms

Inside-Outside algorithm (IO): The IO algorithm works with probability values of words in a learning sample, probabilities of all possible derivations of that words and rule usage for each such derivation. Therefore we have to modify the algorithm to provide all necessary information. Now $P[i, j, k].prob$ represents the probability of deriving a subsequence $a_i a_{i+j-1}$ from a non-terminal S_k .

$P[i, j, k].cnt$ is a count of all possible splittings (to 2 substrings). $P[i, j, k].X[m]$ stores triples referencing all possible substrings; $m \in [1, cnt]$.

1. **set** all elements of P as follows: $P[i, j, k].prob:=0$; $P[i, j, k].cnt:=0$;
2. **for** each $i \in [1, n]$ **for** each $S_j \rightarrow a_i$ rule **set**
 $\{ P[i, 1, j].prob := P(S_j \rightarrow a_i); P[i, 1, j].cnt := 1 \}$;
3. **for** each $i \in [2, n]$ **for** each $j \in [1, n - i + 1]$ **for** each $k \in [1, i - 1]$
for each $S_A \rightarrow S_B S_C$ rule
if $P[j, k, B].prob * P[j + k, i - k, C].prob > 0$ **then**
 $\{ P[j, i, A].cnt := P[j, i, A].cnt + 1;$
 $P[j, i, A].prob := P[j, i, A].prob +$
 $\quad + P[j, k, B].prob * P[j + k, i - k, C].prob * P(S_A \rightarrow S_B S_C);$
 $P[j, i, A].X[P[j, i, A].cnt] := (k, B, C); \}$

CYK algorithm works in polynomial time, but the recalculation of probabilities for the rules is exponential (as we have previously shown that the number of all possible word derivations in a CFG to CNF is exponential), therefore processing all information using this algorithm is exponential.

4.3 Modification of CYK Algorithm for RNA Grammar

Note, that the algorithm could be easily adopted to Viterbi learning. It suffices to store the information about the most probable derivation in the P array (splitting into 2 substrings, two non-terminals and so on). In the third step we only compare $P[j, i, A].prob$ and the value of $P[j, k, B].prob * P[j + k, i - k, C].prob * P(S_A \rightarrow S_B S_C)$. If the latter is greater it is necessary to change $P[j, i, A]$.

Less memory: Nonterminals are divided into two groups. The first one consists of those, which are on the left-hand-side of exactly one grammar rule and are rewritten to a terminal by this rule. Other nonterminals belong to the second group. If k is an index of some first-group nonterminal, the cell $P[i, j, k]$ for $j \neq 1$ doesn't matter. While processing of the string, length n , we can save $n * (n - 1)$ fields for every rule from the first group. RNA grammar constructed according to the method has at least four such nonterminals, which for a word of length 3000 means, the array P will be smaller by almost 36 millions of fields.

Less time: The basic algorithm works with two groups of rules: those rewriting a nonterminal to terminal (in the second step) and the others - rewriting a nonterminal to two nonterminals (in the third step). However, if there is a rule that rewrites a nonterminal to two nonterminals and the first of them belongs to the first group it is not necessary to consider all possible splittings into two substrings in the third step. Therefore we divide the rules into four groups:

1. Rules rewriting a nonterminal to terminal(s).
2. Rules with two nonterminals on the left-hand-side, the first of them belonging into the first group of non-terminals
3. Rules with two nonterminals on the left-hand-side, the second of them belonging into the first group of non-terminals
4. Other rules

In the third step of the algorithm for given i, j and the rule $S_A \rightarrow S_B S_C$ belonging to the second group of rules it is sufficient to consider only one splitting into two substring representing derivation of the first symbol of the word from the nonterminal S_B belonging to the first group of nonterminals and derivations of the rest of the word from the nonterminal S_C . For given i, j and a rule from the second group we spare $i - 2$ iterations. The exact number of operations for each iteration depends on the choice of algorithm for probabilistic estimate. Similarly for the third-group rule $S_A \rightarrow S_B S_C$ we do not need to consider for given i and j all k from 1 to $i - 1$. Derivation of the word $a_j \dots a_{j+i-1}$ using the chosen rule is possible iff the first $i - 1$ symbols are derived from nonterminal S_B and the last symbol of the word is hence derived from the nonterminal S_C . Again we spare $i - 2$ operations. Only for the rules from the fourth group it is necessary to consider all possible splittings. Using the proposed changes it is possible for each rule from the second and the third group to spare $\sum_{i=2}^n (n - i + 1) \cdot (i - 2)$ iterations, where n is a length of a word which makes 200533704 operations less for the length 3000.

Even less memory: If some of the cells stored "zero" probability information in the original algorithm, it could have two different interpretations: (1) The substring represented by this cell is not derivable from the corresponding nonterminal. (2) The mentioned derivation is possible but with zero probability. This situation can happen when we initialize all cells of the array P in the first step. If we omit the first step it is possible (in the second step) to initialize all the cells of P handling the rules from the first group before they are assigned the proper values. Hence the third step of the algorithm should be modified for i, j, k and a rule $S_A \rightarrow S_B S_C$ as follows:

```

if  $P[j, k, B]$  and  $P[j + k, i - k, C]$  were initialized then
  if  $P[j, i, A]$  was not initialized then {initialize  $P[j, i, A]$ ; set  $P[j, i, A]$ }
  else set  $P[j, i, A]$ ;
    
```

Setting of a cell depends on a choice of a probability estimation. The exact amount of memory spared by this modification depends on the given grammar, on an input word and on a chosen probabilistic estimation.

5 Results of Application

5.1 tRNA Grammar Construction

After consideration of structure, length of RNA strings classes and complexity (space and time) of the probabilistic estimates it is possible to construct a grammar which models language of tRNA strings (modelling mRNA cannot be effectively done by CFG as they contain knots and are very long). The grammar contains terminals a, c, g, u and nonterminals S, X, M, V, L . The nonterminals represent patterns of the tRNA structure. It is shown in the Fig. [11](#)

Some separate nucleotids may be attached to the ends 3' and 5' of the tRNA molecule which can easily be covered by these rules: $S \rightarrow aS|cS|gS|uS$ for 5'

termination and $S \rightarrow Sa|Sc|Sg|Su$ for 3'. The nucleotids that form the basic pairs follow hence the grammar contains a rule $S \rightarrow X$. Stem can be modelled by $X \rightarrow aXu|cXg|gXc|uXa$ and its transition to the first part of multiloop by $X \rightarrow M_1$ rule. In case that M_1 or M_1 and M_2 are of zero length we add rules $X \rightarrow VM_2$ and $X \rightarrow VVM_3$ for the nonterminal X . Some sources [1], [4], [12] state that the third part of multiloops is functionally similar to the second loop. Therefore we assume nonemptiness of the substring M_3 .

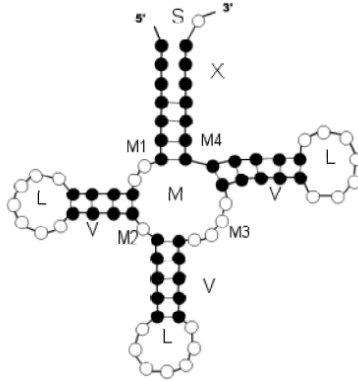


Fig. 1. S - start symbol, X - stem, M - multiloop, V - hairpin, L - loop

In the first step we describe the rules for parts of multiloops. It may consist of some nucleotids followed by hairpin pattern. So the nonterminal M_1 is rewritten by the rules $M_1 \rightarrow aM_1|cM_1|gM_1|uM_1$, $M_1 \rightarrow VM_2$ and $M_1 \rightarrow VVM_3$ in case the second part of multiloops is of length 0. Following the same arguments as in case of the nonterminal M_1 we add rules $M_2 \rightarrow aM_2|cM_2|gM_2|uM_2$, $M_2 \rightarrow VM_3$, $M_3 \rightarrow aM_3|cM_3|gM_3|uM_3$, $M_3 \rightarrow VM_4$ and $M_3 \rightarrow V$ for nonterminals M_2 and M_3 . The last part of multiloops must be after some time rewritten to a terminal which is covered by these rules: $M_4 \rightarrow aM_4|cM_4|gM_4|uM_4$ and $M_4 \rightarrow a|c|g|u$.

The pattern of hairpin is coverable by:

$$V \rightarrow aVu|cVg|gVu|uVa, V \rightarrow aL|cL|gL|uL, L \rightarrow aL|cL|gL|uL, L \rightarrow a|c|g|u.$$

This grammar is CFG, but if we want to use the CYK algorithm it must be transformed to CNF using the aforementioned process. This grammar in CNF models basic pairs only. The alternative pairs can be added by extending the grammar using rules rewriting nonterminals S and X to strings $AY_c|CY_a|GY_u|UY_g$ and nonterminals M_3 and V to $AW_c|CW_a|GW_u|UW_g$. The new nonterminals Y_i and W_i , $i \in \{a, c, g, u\}$ can be rewritten as follows: $Y_u \rightarrow XU$, $Y_g \rightarrow XG$, $Y_c \rightarrow XC$, $Y_a \rightarrow XA$, $W_u \rightarrow VU$, $W_g \rightarrow VG$, $W_c \rightarrow VC$, $W_a \rightarrow VA$.

In order to explore properties of loops the nonterminal L can be substituted by a sequence of nonterminals L_1, \dots, L_n where n is the longest possible loop in tRNA pattern. For $i = 1, \dots, n - 1$ the grammar contains the rules $L_i \rightarrow aL_{i+1}|cL_{i+1}|gL_{i+1}|uL_{i+1}$ and $L_i \rightarrow a|c|g|u$, for the nonterminal L_n only the rules

rewriting it to a terminal: $L_n \rightarrow a|c|g|u$. Further extension enables us to investigate the size of loops, measures of stems, or size of the parts of multiloops. In the grammar, the number of the prepared rules was 165. After the computation of the rules probabilities we removed 57 rules with the probability 0. The result of SCFG after 70 iterations under the normal distribution are in the Table [1](#).

Table 1. SCFG for *Acinetobacter* sp. bacteria.

| Rule | Prob. | Rule | Prob. | Rule | Prob. | Rule | Prob. |
|--------------------|-------|--------------------|-------|--------------------|-------|--------------------|-------|
| $S \rightarrow US$ | 0.012 | $S \rightarrow SA$ | 0.198 | $S \rightarrow SC$ | 0.064 | $S \rightarrow SG$ | 0.096 |
| $S \rightarrow SU$ | 0.141 | $S \rightarrow AB$ | 0.064 | $S \rightarrow CD$ | 0.173 | $S \rightarrow GE$ | 0.230 |
| $S \rightarrow UF$ | 0.019 | $X \rightarrow AB$ | 0.122 | $X \rightarrow CD$ | 0.242 | $X \rightarrow GE$ | 0.370 |
| $X \rightarrow UF$ | 0.080 | $X \rightarrow A1$ | 0.035 | $X \rightarrow U1$ | 0.114 | $X \rightarrow Cd$ | 0.007 |
| $X \rightarrow Ge$ | 0.019 | $X \rightarrow Uf$ | 0.005 | $1 \rightarrow A1$ | 0.238 | $1 \rightarrow C1$ | 0.207 |
| $1 \rightarrow G1$ | 0.274 | $1 \rightarrow U1$ | 0.228 | $1 \rightarrow V2$ | 0.006 | $1 \rightarrow VW$ | 0.044 |
| $2 \rightarrow A2$ | 0.270 | $2 \rightarrow C2$ | 0.305 | $2 \rightarrow G2$ | 0.223 | $2 \rightarrow U2$ | 0.094 |
| $2 \rightarrow V3$ | 0.105 | $3 \rightarrow U3$ | 0.037 | $3 \rightarrow V4$ | 0.139 | $3 \rightarrow AH$ | 0.126 |
| $3 \rightarrow CI$ | 0.139 | $3 \rightarrow GJ$ | 0.240 | $3 \rightarrow Ci$ | 0.253 | $3 \rightarrow Gj$ | 0.063 |
| $4 \rightarrow C4$ | 0.368 | $4 \rightarrow U4$ | 0.052 | $4 \rightarrow a$ | 0.526 | $4 \rightarrow u$ | 0.052 |
| $V \rightarrow AH$ | 0.095 | $V \rightarrow CI$ | 0.188 | $V \rightarrow GJ$ | 0.244 | $V \rightarrow UK$ | 0.083 |
| $V \rightarrow AL$ | 0.027 | $V \rightarrow CL$ | 0.023 | $V \rightarrow GL$ | 0.053 | $V \rightarrow UL$ | 0.102 |
| $V \rightarrow Ah$ | 0.037 | $V \rightarrow Ci$ | 0.077 | $V \rightarrow Gj$ | 0.039 | $V \rightarrow Uk$ | 0.028 |
| $L \rightarrow AM$ | 0.135 | $L \rightarrow CM$ | 0.043 | $L \rightarrow GM$ | 0.149 | $L \rightarrow UM$ | 0.412 |
| $U \rightarrow u$ | 1.0 | $L \rightarrow c$ | 0.043 | $L \rightarrow g$ | 0.214 | $M \rightarrow AN$ | 0.053 |
| $M \rightarrow CN$ | 0.455 | $M \rightarrow GN$ | 0.047 | $M \rightarrow UN$ | 0.153 | $M \rightarrow a$ | 0.017 |
| $M \rightarrow c$ | 0.053 | $M \rightarrow u$ | 0.218 | $N \rightarrow AO$ | 0.2 | $N \rightarrow CO$ | 0.083 |
| $N \rightarrow GO$ | 0.575 | $N \rightarrow UO$ | 0.141 | $O \rightarrow AP$ | 0.658 | $O \rightarrow CP$ | 0.208 |
| $O \rightarrow GP$ | 0.125 | $O \rightarrow UP$ | 0.008 | $P \rightarrow AR$ | 0.391 | $P \rightarrow CR$ | 0.033 |
| $P \rightarrow GR$ | 0.325 | $P \rightarrow UR$ | 0.25 | $R \rightarrow AT$ | 0.041 | $R \rightarrow a$ | 0.116 |
| $R \rightarrow c$ | 0.266 | $R \rightarrow u$ | 0.575 | $T \rightarrow GY$ | 1.0 | $Y \rightarrow u$ | 1.0 |
| $Z \rightarrow a$ | 0.25 | $Z \rightarrow c$ | 0.25 | $Z \rightarrow g$ | 0.25 | $Z \rightarrow u$ | 0.25 |
| $B \rightarrow XU$ | 1.0 | $D \rightarrow XG$ | 1.0 | $E \rightarrow XC$ | 1.0 | $F \rightarrow XA$ | 1.0 |
| $b \rightarrow XC$ | 1.0 | $d \rightarrow XA$ | 1.0 | $e \rightarrow XU$ | 1.0 | $f \rightarrow XG$ | 1.0 |
| $W \rightarrow V3$ | 1.0 | $H \rightarrow VU$ | 1.0 | $I \rightarrow VG$ | 1.0 | $J \rightarrow VC$ | 1.0 |
| $K \rightarrow VA$ | 1.0 | $h \rightarrow VC$ | 1.0 | $i \rightarrow VA$ | 1.0 | $j \rightarrow VU$ | 1.0 |
| $k \rightarrow VG$ | 1.0 | $A \rightarrow a$ | 1.0 | $C \rightarrow c$ | 1.0 | $G \rightarrow g$ | 1.0 |

5.2 Probabilities of Rules

Our aim was to estimate the probability values of the grammar rules using a training sample. We have chosen the Viterbi learning algorithm, because of its lower time complexity. The training sample consisted of tRNA strings of *Acinetobacter* sp. bacteria. ADP1 Complete genetic information was acquired from http://www.sanger.ac.uk/cgi-bin/Rfam/genome_dist.pl. The aim of the learning process was to acquire the probability vector for the proposed grammar. This vector would enable the grammar to generate tRNA chains of this cell only. We extended the basic grammar by adding rules admitting alternative pairs

and exploring the size of loops in general. Learning algorithm was applied until the probability vector was not equal to the previous iteration's one. We compared learning process for two distinct initial settings of the probability vector (equal probabilities for the rules rewriting the same nonterminal vs. lower values for the rules modelling alternative pairs). In both cases the resulting setting of the probability vector had similar properties. After generating words we found out that the grammars are too general to satisfactorily cope with tRNA chains of one organism. Further research can therefore be focused on learning tRNA chains for some species (i.e. for all bacteria). It is also possible to try to extend the grammar by some new rules. The grammar we gained can describe some properties of one class of tRNA chains. Beforemost it is the information about the most probable length of a loop contained in the chain (in our case it was 6 nucleotids). Also it is possible to get occurrence ratio of the nucleotid in a given pattern.

6 Conclusions

We extended CYK algorithm for use in the area of probabilistic estimations for stochastic grammars. We proposed a modification of the algorithm for learning algorithm of stochastic grammars modelling RNA strings which led to space and time complexity reduction. We also proposed a modification capable of deriving the most probable derivations of a word in Viterbi learning (even for words with long derivations. We built the grammar which covers tRNA chains properties. In the experiment [7], we tested the learning using the Viterbi algorithm on the sample representing real biological data of *Acinetobacter* sp. bacterium.

References

1. Baldi, P. and Brunak, S.: *Bioinformatics, The machine learning approach*, MIT Press, Cambridge, 2001.
2. Črepinšek, M., Merník, M., Javed, F., Bryantt, B. R. and Sprague, A.: *Extracting Grammar from Programs: Evolutionary Approach*, ACM SIGPLAN Notices, Vol. 40, No. 4, 2005, p. 39–46,
3. Gold, E. M.: *Language Identification in the Limit, Information and Control*, Academic Press Inc., 10, 1967, p. 447–474.
4. Jones, N. C. and Pevzner, P. A.: *An Introduction to Bioinformatics Algorithms*, The MIT Press, Cambridge, 2004.
5. Jurafsky, D. and Martin, J. H.: *Speech and Language processing*, Prentice Hall, New Jersey 07458, 2002.
6. Linares, D., Benedí, J. M. and Sánchez, J. A.: *A Hybrid Language model based on a Combinations of N-Grams and Stochastic Context-Free Grammars*, ACM Trans. on Asian Language Information Processing, Vol. 3, No. 2, 2004, p. 113–127.
7. Lengeňová, H.: *Stochastic Grammars and Linguistics*, RNDr. Thesis, P. J. Šafárik University, Košice, 2005 (in Slovak).
8. Rivas, E. and Eddy, S. R.: *The language of RNA: a formal grammar that includes pseudoknots*, *Bioinformatics*, Vol. 16, No. 4, 2000, p.334–340.

9. Sanchez, J. A. and Benedi, J. M.: Consistency of Stochastic Context-Free Grammars from Probabilistic Estimation Based on Growth Transformation, *IEEE Trans. on pattern Analysis and Machine Intelligence*, Vol. 19, No. 9, 1997, p. 1052–1056.
10. Sakakibara, Y. and Muramatsu, H.: Efficient learning of context-free grammars from partially structured examples, *Proc. of the 5th Int. Colloquium on Grammatical Inference and Applications, ICG'00, LNAI*, Vol. 1981, 2000, p. 229–240.
11. Sakakibara, Y.: Grammatical Inference in Bioinformatics, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 7, July 2005, p. 1051– 1062.
12. Westhead, D. R. and Parish, J. H.: *Bioinformatics*, BIOS Scientific Publisher Ltd, Oxford, 2002.

Modeling Human Performance in Two Player Zero Sum Games Using Kelly Criterion

Rafal Lopatka and Andrzej Dzielinski

Institute of Control & Industrial Electronics
Warsaw University of Technology
00-662 Warszawa, ul. Koszykowa 75
{lopatkar, adziel}@isep.pw.edu.pl

Abstract. The paper presents a new way of modeling the human performance in two player games using a rating system based on Kelly Criterion which is often utilized for gambling and financial engineering. The advantage of the proposed system is the ability to assess playing strength based on both the final outcome of the game and the style of play. The last aspect of the rating assessment system is novel compared to the rating systems developed so far (like ELO, Bradley-Terry, etc.). Another advantage of the proposed method is the tackling of the problem of drawn games. To the very best authors knowledge the approach presented below is a relatively new look at the problem of playing strength assessment based on probability theory. The paper presents and discusses few illustrative examples.

1 Introduction

Modeling of human behavior is a problem of great difficulty and importance in many areas of life. The decision making support systems are but one of the examples where human behavior plays a significant role and having a valid model of it is of utmost importance. In the paper we consider one of the cases of decision making i.e., two player zero sum games, such as a game of chess, which sets the context of the following considerations on modeling (measurement) of human player performance. The problem of comparison of playing strength of human players is almost as old as the games themselves. Comparing playing strength is very difficult because various playing strategies and styles must be considered with no sharp criteria. In the case of human players the task is even more complex due to the strong influence of player's mentality, shape, age or even factors unexplainable by the players themselves. The latter may be illustrated very well with an example from a recent chess event. On November 27 2006 current world champion Vladimir Kramnik played his second game of the match vs computer chess program Deep Fritz. According to judgement of the chess experts up to the 34th move, the best human player was playing very well, even brilliantly. The position was as given in Fig. 1 with Kramnik to move.

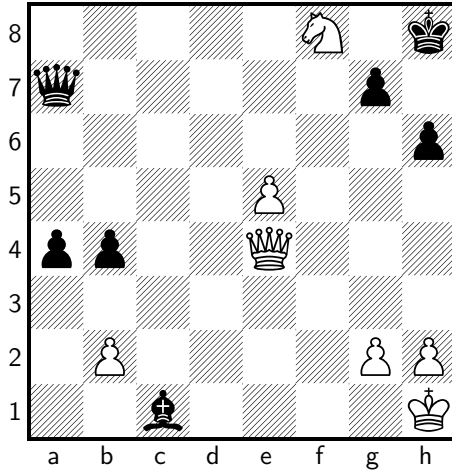


Fig. 1. Chessboard position at which Kramnik played a surprisingly bad move

Kramnik played 34...Qe3?? allowing Deep Fritz to checkmate on the 35th move. Hard to believe, but true. After the game Kramnik stated:

I'm also shocked by what happened. I can't explain it. My position was excellent; I felt good, and wasn't even tired.

The words and the performance of the world champion prove that human behavior can be unpredictable even at the highest level of proficiency. It would be very difficult to create a deterministic model covering all factors influencing humans behavior. Such a task is very hard, if possible at all. This is probably the reason for great success of the probability theory in the field since it provides a framework to cover (better or worse) the influence of most of the possible variables having influence on the players performance quality.

It is in general acceptable to assume that the history of probability based rating systems starts in the 60's of 20th century with the approach proposed by Bradley and Terry in [1]. Since throughout the paper (due to personal preferences) we use the game of chess as an example of two player zero sum game, an extension of hungarian mathematician Arpad Elo [2] to Bradley and Terry rating system will become a subject of interest. However, all remarks formulated with respect to Elo system apply to the original idea of Bradley and Terry.

Following Bradley and Terry, Elo proposed to assign chess player a number representing his/her strength of play. Throughout the years biggest chess federations accepted Elo rating system and nowadays it has become the most popular of all ratings in chess world. Despite of its popularity the Elo system (as well as its originator) has one major drawback. It is draw handling. With huge simplification one can say tha such a result of the game is theoretically quite inconvenient as the rating change mainly depends on difference in wins and loses. Elo system does not efficiently answer the question how to handle draws. Many tricks and

some more sophisticated solutions have been proposed in order to tackle this problem. However, the problem has not been solved entirely.

This paper aims to provide an alternative rating system utilizing Kelly Criterion presented in [3], which gives a clear solution of this problem. It will be shown that Kelly Criterion allows to take a slightly different look at the two player zero sum games simultaneously giving some kind of measure of style of play which is not possible to obtain in the case of rating systems well known so far.

2 Kelly Criterion

Kelly criterion originates in 1956 with the paper [3] where a problem of gambler's money growth is considered and has been successfully adopted for many other applications, especially financial engineering (see [4]). Kelly developed a formula for gambling assuring exponential gambler's wealth grow using Shannon's considerations on transmission channel errors presented in [5]. Shannon defines the rate of transmission over a noisy communication channel in terms of probability theory. Kelly has transferred theoretical model based on probability proposed by Shannon into a gambling system, provided some additional conditions are satisfied.

2.1 Assumptions

Suppose there are two players, say player A and player B, wagering the money on some event. Let player A have some finite amount of money and let the player B be infinitely wealthy. Assume both player's capital is infinitely divisible and player A wins with probability p , player B wins with probability $q = 1 - p$, no ties. The main assumption is, there are some "fair" bets available, i.e. bets having odds consistent with win and lose probabilities.

The matter of identification of such bets is a separate problem. However, it is quite easy to find such events. As an example a Black Jack can be mentioned. This game (counterpart to many other casino games) can be favorable to a player, provided card counting is applied.

2.2 The Criterion

With the above assumptions satisfied, Kelly defines an exponential measure of player's capital growth rate as follows

$$g(f) = p \log(1 + f) + q \log(1 - f) \quad (1)$$

where f is the fraction of a capital wagered by player A on a single trial, p, q state probabilities as described above. For more details see [3,4]. Trivial calculations on (1) lead to a conclusion that optimal growth rate value with respect to fraction bet f takes place for

$$f^* = p - q \quad (2)$$

Kelly criterion implies many important conclusions. Below just two of them are recalled.

1. If we find an event with positive expectation, i.e. $g(f) > 0$ for some f , we for sure can increase our capital to arbitrary amount by betting fraction (2) of bankroll on every trial.
2. The above strategy is optimal. This means that in the sufficiently long time period the player using this strategy will become more wealthy than player using any other strategy while betting on the same event.

Another important remark coming out of the Kelly criterion is on expected time of doubling the bankroll. Since we use Kelly criterion for betting we have exponential growth rate f^* and therefore to double the bankroll we have to perform n trials, where

$$n = \frac{\log 2}{g(f^*)} = \frac{\log 2}{g(p - q)} \quad (3)$$

3 The Link to Ttwo Player Games

In this section let us concentrate our attention on the way of transferring the Kelly criterion results into the field of two player games. In this context the players will become gamblers, the game will become an event, and the capital possessed by both players will be limited to a unit. A single move of the game made by both players can be viewed as single trial in gambling. Both players bet some part of their capital on each trial. If player A makes better (however defined) move than player B, he/she wins a single trial and the capital fraction bet by the opponent. The same takes place if player B wins a single trial. The ultimate goal for both players is to double the bankroll. This seems to be a natural assumption in a zero-sum game, where both players start with a unit capital. In such a case the game ends when either of them doubles his/her bankroll. This means that the other player's capital is zero. Of course, both players chances for victory are not equal and (by assumption) proportional to the rating assigned. Usually player ranked higher is expected to win, so in all further examples the game evaluation and rating changes will always be calculated from winner's point of view. This is also due to the fact that Kelly assumes application of his criterion for positive expectation events only, which in the context of the game is equivalent to winner's point of view.

3.1 Assumptions

We assume that

1. The game is always played to the terminal state (win, lose, draw) without resigning.
2. Each player gets a unit initial capital and tends to double it during the course of the game.
3. Both players play at their best.
4. Players ratings changes compensate each other.
5. Rating changes take place after every single game.

3.2 Expected Time to Win and Rating Changes

Formula (3) gives the expected number of trials necessary to double the possessed capital provided we know probabilities p and q , or equivalently the optimal exponential growth rate $g(p - q)$. In the case of rating system we propose to look at this formula in a little different way. In the light of previously stated assumptions and the result of the game one can ask: *what are the probabilities p and q and the growth rate $g(p - q)$ in (3) for the game finished in n moves?* The answer is simple in the case of optimal exponential growth rate

$$g(p - q) = \frac{\log 2}{n} \tag{4}$$

With the growth rate calculated it is possible to proceed further. Having in mind assumption on optimal play by both players and the relation $p = 1 - q$ we can use (1) to get the following equation

$$g(p - q) = q \log 2q + (1 - q) \log 2(1 - q) \tag{5}$$

which solved with respect to q implies simultaneously p . With these data at hand we can view the game as a probabilistic experiment in which better player (not necessarily the one higher ranked) had a $\max(p, q)$ chances for winning a single trial and the weaker player had a $\min(p, q)$ chances for winning a single trial, respectively.

The recipe for rating change may be as follows. Before the game both players had together some amount of rating points. With assumption 4 the sum of both player’s ratings does not change. However, due to the performance of players during the game the total sum of rating points may split differently from the initial state. We suggest to split the entire number of points proportionally to the probabilities p and q obtained from the above calculations. We get, in this way, differences between initial rating and the rating implied by the result of the game. Having these differences calculated for both players we can change their ratings according to some arbitrary strategy (e.g. we may change the ratings by only 10% of the calculated difference).

Before we proceed to the examples we want to emphasize one important advantage of the proposed solution over existing rating systems: *it is able to reward (or punish) style of play by introducing exponential growth rate into the model.* This will be illustrated with an example below.

Example 1. Few years ago one of the authors had an opportunity and pleasure to participate in a simultaneous game with Polish strongest grandmaster. The grandmaster was ranked that time about 2600 Elo. The estimate of the other player strength was about 1600 Elo. The author lost the game after 32 moves. How did he perform according to the Kelly Criterion? To evaluate this performance it is necessary to calculate the growth rate with (3)

$$g(p - q) = \frac{\log 2}{32} = 0.02166$$

This is the number reflecting how fast the grandmaster was supposed to increase his advantage over the adversary. Now we have to solve the equation

$$0.02166 = q \log 2q + (1 - q) \log 2(1 - q)$$

to get values of $p \approx 0.6025$ and $q \approx 0.3975$. This means with every move author had 39.75% probability for improving the position on the board and 60.25% probability to make it weaker, opposite to the opponent.

Before the game we had in total $2600 + 1600 = 4200$ rating points. With new values of p and q we can make a new split of this number reflecting our performance in the played game. The new rating of grandmaster would be

$$\max(p, q) * 4200 \approx 2530$$

since he was the winner and the new rating for the author would be

$$\min(p, q) * 4200 \approx 1670$$

since he lost the game. However, the above calculations show that the grandmaster played weaker than 2600 points rated player or his adversary resited longer than 1600 points ranked player should have resited. Despite of losing the game according to the Kelly Criterion the loser should be rewarded for resiting the grandmaster more than expected and the grandmaster would be punished for playing not well enough against 1600 points rated player. This is an example of how the Kelly Criterion evaluates the style of play. In this case the grandmaster's rating is to be decreased by 70 rating points, opposite to the rating of the other player. How much finally the calculated difference influences both players ratings is an arbitrary choice. With the suggestion was made above the new ratings would be 2593 and 1607 for the grandmaster and the author, respectively.

Now we can try to analyze how well the grandmaster should play to avoid decreasing his rating after the game vs such a weak player like in the example. In order to answer this question we quickly can calculate the grandmasters probability p he wins a single trial in our probabilistic experiment with initial grandmaster's rating,

$$p \geq \frac{2600}{4200} \approx 0.62$$

With this value we can estimate the player ranked 2600 not willing to decrease his rating should beat player ranked 1600 in less than

$$n = \frac{\log 2}{0.029} \approx 24$$

moves since

$$g(p - q) = 0.62 \log(1 + 0.62 - 0.38) + 0.38 \log(1 - 0.62 + 0.38) = 0.029$$

To make our considerations more complete consider an abstract case when the game is lost by the grandmaster.

Example 2. Suppose the player managed to win the game vs the grandmaster. The calculation proceeds the same way as in the example above and the only difference is that the player and the grandmaster exchange their roles. This means the player gets $p = 60.25\%$ share of total ratings points and the grandmaster gets $q = 39.75\%$ of the total. If so, the player played like a one ranked 2530 and the opponent like a player ranked 1670. The rating change for the player would be $2530 - 1600 = 930$ and -930 for the grandmaster losing the game. With the suggestion assumed in the previous example the player would get his rating raised to 1693 rating points and the opponent would get his rating down to 2507.

Two examples presented so far show how to deal with wins and loses in the framework of the rating system based on Kelly Criterion. In the next section the recipe for draw handling is presented which is an inconvenient, from theoretical point of view, game result for Elo rating system and other similar rating systems based on Bradley and Terry idea.

3.3 Ties

Ties are hardest problem for so far known rating systems. In fact none of these deals with this result of the game directly, but with some tricks which mainly tend to replace draws with equivalent, in the sense of rating points, number of wins and loses. As an example may be the proposition of exchanging two draws got by the player against the same opponent with a win and lose. Many more can be cited but this is not the point of this section. Here we want to show how Kelly Criterion provides an easy way to handle draws. It is easy to figure it out when one notices two basic facts:

- both players performed equally well and this transferred into probability language of Kelly Criterion means $p = q = 0.5$ regardless of the number of moves,
- none of the players managed to double the bankroll so $g(p - q) = 0$.

The total sum of rating points is spread equally among players and the rest of the procedure remains the same as in case of a victory.

Example 3. Assume the player managed to draw the game vs the grandmaster. Since both played equally strong both got half of total points available before the game, which means 2100 rating points for each of them, as well as the difference in rating equal to 500 rating points. Therefore the players rating after the game would be increased to 1650 where the opponent's rating would be decreased to 2550.

Worth of explicit formulation is that the Kelly Criterion rating based system for draws forfeits its style evaluation property since number of moves play no role in calculations.

4 Conclusions

To summarize the above short considerations let us point out few remarks on the new rating system. First of all it is worth to notice that the system is capable to reflect style of play to rating change (see the first example) by incorporating number of moves into calculations. Only in case of a draw the amount of moves in the game has no meaning. In this framework better player can get his rating down in case of victory in more moves than expected. Compare lost game vs Kasparov in 18 moves and the game lost in, say, 70 moves. Most players would agree that defeat in 70 moves vs Kasparov would indicate tough battle on the board as opposed to defeat in 18 moves. Other argument might be that the top ranked players used to lose to Kasparov faster than in 70 moves, but never in 18 moves. The Kelly criterion based rating system is able to catch that difference. Another advantage of the proposed solution is improved draw handling without various tricks used in other rating systems. Also it is worth mentioning that basing on the Kelly criterion it is possible to construct a rating system not only when a single game is played but also for a tournament. In such a case several players play against each other and the rating system can take into account their respective results. This is definitely more complicated because during a tournament one may come across some surprising results affecting the players' ratings more than the results which were expected. The drawbacks of this system can be the necessity of rating calculation after every single game and its quite restrictive character. The second can be handled by application of some rule scaling rating changes. However, it depends on arbitrary choice.

References

1. Bradley R.A., T.M.: The rank analysis of incomplete block designs. 1. the method of paired comparisons. *Biometrika* **39** (1952)
2. A.E., E.: The rating of chess players past and present. Arco Publishing, New York (1978)
3. J.L., K.: A new interpretation of information rate. *Bell System Technical Journal* **35** (1956)
4. E.O., T.: The Kelly criterion in black jack, sports betting, and the stock market. *The 10th International Conference on Gambling and Risk Taking* (1997)
5. C.E., S.: A mathematical theory of communication. *B.S.T.J.* **27** (1948)

No-Regret Boosting^{*}

Anna Gambin¹ and Ewa Szczurek^{1,2}

¹ Inst. of Informatics, Warsaw University, Banacha 2, 02-097 Warsaw, Poland

² Max Planck Inst. for Molecular Genetics, Ihnestrasse 73, 14195 Berlin, Germany
aniag@mimuw.edu.pl, szczurek@mimuw.edu.pl

Abstract. Following [4], we analyze boosting from a game-theoretic perspective. We define a wide class of boosting classification algorithms called H-boosting methods, which are based on Hannan-consistent game playing strategies. These strategies tend to minimize the regret of a player, i.e. are able to minimize the difference between its expected cumulative loss and the cumulative loss achievable using the single best strategy. The “weighted majority” boosting algorithm [4] is proved to belong to the class of H-boosting procedures. A new boosting algorithm is proposed, as an another example of such a regret-minimizing method.

1 Introduction

Motivation and related work. Boosting is a “meta” classification method: it improves performance of any given learning algorithm which generates classifiers achieving accuracy only a little better than random guessing. According to Breiman [1], the theory behind the success of adaptive reweighing and combining algorithms such as Adaboost has not yet been well understood. This work analyzes boosting from a game-theoretic perspective. The close connections between game theory and boosting were first studied in [4], where boosting was implemented using the Littlestone and Warmuth’s “weighted majority” algorithm [6].

Our approach. A crucial concept in analysis of the game playing procedures considered is the notion of regret. Intuitively, regret is a measure of how much the player “regrets” playing his adaptive strategy instead of choosing any other one. We focus our interest on the procedures which by adaptive learning tend to minimize the regret of the row player (learner) playing against the column player (environment). These procedures are called no-regret (or regret-matching) strategies. If the strategy is no-regret regardless of the adaptive procedure chosen by the environment, it is called Hannan-consistent [5].

By formulating boosting as a repeated game where one player makes a selection from instances in the training set and the other from a set of hypothesis produced by the base classifiers, we can evaluate resulting algorithm in a game-theoretic context. In this setting boosting performance is maximized by the second player, which by employing an appropriate strategy tries to maximize loss

* The research described in this paper was partially supported by Polish Ministry of Education and Science grant 4T11C04425.

of the first one. We explored a number of adaptive strategies considered by [2] and analyzed them with respect to their applicability to the boosting problem.

Our results. Following [2] we consider two (exponential and polynomial) general classes of adaptive strategies. Algorithms of both these classes were proved to be Hannan-consistent. We define a new class of H-boosting procedures, comprising E-boosting and P-boosting methods which can be implemented as repeated game playing using those adaptive strategies. A new boosting algorithm AdaBoost.P based on a P-boosting procedure is proposed. The algorithm derived theoretically as a regret-minimizing procedure, implemented in the R environment [7] on the basis of the boosting package `ada` [3], shows high performance in practice.

2 Preliminaries

We follow the notation and definitions given in [4]. Only two-person, zero-sum games in normal form are considered. In this case, a game can be defined by a finite matrix \mathbf{M} . The two opponents are referred to as the row and the column player. $\mathbf{M}^{i,j}$ is the loss the row player suffers playing i when the column player plays j . For the sake of simplicity, we assume $\mathbf{M}^{i,j} \in [0, 1]$, where $i \in 1 \dots n$ and $j \in 1 \dots m$.

In a randomized setting, to play a game, the row player chooses a distribution $\mathbf{P} = (\mathbf{P}^1, \dots, \mathbf{P}^n)$ over the rows of \mathbf{M} (called a *mixed strategy*), and the column player chooses a distribution $\mathbf{Q} = (\mathbf{Q}^1, \dots, \mathbf{Q}^m)$ over columns. The row player's expected loss (referred to simply as loss) is computed as $\mathbf{M}(\mathbf{P}, \mathbf{Q}) = \sum_{i,j} \mathbf{P}^i \mathbf{M}^{i,j} \mathbf{Q}^j = \mathbf{P}^T \mathbf{M} \mathbf{Q}$.

If the row player chooses a distribution \mathbf{P} but the column player chooses a single column j (called a *pure strategy*), then the (expected) loss is $\mathbf{M}(\mathbf{P}, j) = \sum_i \mathbf{P}^i \mathbf{M}^{i,j}$. The notation $\mathbf{M}(i, \mathbf{Q})$ is defined analogously.

We assume the players move in sequential manner. That is, the column player chooses its strategy \mathbf{Q} after the row player has chosen and announced its strategy \mathbf{P} . In the zero-sum game context, the row player minimizes his loss. A mixed strategy realizing this minimum $\mathbf{P}^* = \arg \min_{\mathbf{P}} (\max_{\mathbf{Q}} \mathbf{M}(\mathbf{P}, \mathbf{Q}))$ is called a *minmax strategy*.

Suppose now that the column player plays first choosing \mathbf{Q} and the row player can choose its play with the benefit of knowing \mathbf{Q} in advance. Here a mixed strategy optimal for the column player (whose goal is to maximize the row player's loss) $\mathbf{Q}^* = \arg \max_{\mathbf{Q}} (\min_{\mathbf{P}} \mathbf{M}(\mathbf{P}, \mathbf{Q}))$ is called a *maxmin strategy*.

By the von Neumann's MAXMIN Theorem the loss of the row player playing a minmax strategy is equal to the loss of the column player playing the maxmin strategy and is called a *game value*. Thus, classical game theory says that given a zero-sum game \mathbf{M} , one should play using a minmax (maxmin) strategy.

In our setting, the game is played repeatedly in a sequence of t rounds, with values of the game matrix \mathbf{M} hidden from the players. We refer to the row player as the *learner* and the column player as the *environment*. Consecutive steps of the game, reveal information on \mathbf{M} gradually. The decisions made are available to the players immediately after each step. On round $s = 1, \dots, t$:

1. The learner chooses mixed strategy \mathbf{P}_s .
2. The environment chooses mixed strategy \mathbf{Q}_s .
3. For all $i = 1, \dots, n$ the learner is permitted to observe the loss $\mathbf{M}(i, \mathbf{Q}_s)$ (i.e. the loss it would have suffered had it played using pure strategy i).
4. The learner suffers loss $\mathbf{M}(\mathbf{P}_s, \mathbf{Q}_s)$.

Then it is natural to ask if the row player is able to learn to play well against the particular opponent that it is facing and to find an approximate minmax (maxmin) strategy in many iterations, basing on the repeated game history and the information available. The learner is to adopt such a procedure of choosing his mixed strategies P_1, \dots, P_t in the series of rounds against the sequence of plays Q_1, \dots, Q_t chosen by the environment that will assure its cumulative loss $\sum_{s=1}^t \mathbf{M}(\mathbf{P}_s, \mathbf{Q}_s)$ is “not much worse” than the loss of the best strategy in hindsight, i.e. $\min_{\mathbf{P}} \sum_{s=1}^t \mathbf{M}(\mathbf{P}, \mathbf{Q}_s)$. The procedure the learner is to adopt is called an *adaptive strategy*. All the adaptive strategies are based on the idea of assigning a higher probability to better-performing actions.

Algorithm LW. Based on Littlestone and Warmuth’s “weighted majority” algorithm [6], it was presented in the repeated play context in [4]. The algorithm proceeds as follows. The learner maintains nonnegative weights on the rows of \mathbf{M} . Let \mathbf{w}_t^i denote the weight at time t on row i . Initially, all the weights are set to unity. On each round t , the learner computes mixed strategy \mathbf{P}_t by normalizing the weights so that they sum up to unity to form a probability distribution:

$$\forall_{i=1, \dots, n} \quad \mathbf{P}_t^i = \frac{\mathbf{w}_{t-1}^i}{\sum_j \mathbf{w}_{t-1}^j} \tag{1}$$

Then, given $\mathbf{M}(i, \mathbf{Q}_t)$ for each i , the learner updates the weights by a simple multiplicative rule, decreasing the weight on a pure strategy with a higher loss in the previous round: $\mathbf{w}_t^i = \mathbf{w}_{t-1}^i \cdot \beta^{\mathbf{M}(i, \mathbf{Q}_t)}$. Here $\beta \in [0, 1)$ is a parameter of the algorithm. By unfolding this simple recurrence, we get $\mathbf{w}_t^i = \beta^{\sum_{s=1}^t \mathbf{M}(i, \mathbf{Q}_s)}$, thus obtaining the following value of the i -th component of P_t in Eq. (1):

$$\mathbf{P}_t^i = \frac{\beta^{\sum_{s=1}^{t-1} \mathbf{M}(i, \mathbf{Q}_s)}}{\sum_j \beta^{\sum_{s=1}^{t-1} \mathbf{M}(j, \mathbf{Q}_s)}}, \tag{2}$$

Theorem 1 (Freund and Schapire, 1996). *For any matrix \mathbf{M} with n rows and entries in $[0, 1]$, and for any sequence of mixed strategies Q_1, \dots, Q_t played by the environment, the sequence of mixed strategies P_1, \dots, P_t produced by algorithm LW with parameter $\beta = \frac{1}{1 + \sqrt{\frac{2 \ln n}{t}}}$ satisfies:*

$$\frac{1}{t} \sum_{s=1}^t M(P_s, Q_s) \leq \min_P \frac{1}{t} \sum_{s=1}^t M(P, Q_s) + \Delta_t^{LW}, \tag{3}$$

where $\Delta_t^{LW} = \sqrt{\frac{2 \ln n}{t}} + \frac{\ln n}{t}$.

Regret. The notion of regret gives a good intuitive understanding of the adaptive strategies that can be proposed for the learner to employ in a repeated game.

Definition 1 (Regret). *The regret the player feels for playing strategy \mathbf{P}_t at time t is a vector \mathbf{r}_t , whose i -th component corresponds to regret felt for not choosing the i -th pure strategy and is represented by: $\mathbf{r}_t^i = \mathbf{M}(\mathbf{P}_t, \mathbf{Q}_t) - \mathbf{M}(i, \mathbf{Q}_t)$, the difference of the loss the player suffered and the loss it would have suffered had it chosen i .*

Definition 2 (Cumulative regret). *Cumulative regret is defined as a vector \mathbf{R}_t whose i -th component is defined as: $\mathbf{R}_t^i = \sum_{s=1}^t \mathbf{r}_s^i$.*

An important class of adaptive strategies, called *Hannan-consistent*, was distinguished by Hannan [5]. As time grows to infinity, regardless of what the opponent’s actions are, the average per-round loss of the learner playing a Hannan-consistent strategy exceeds that of the best possible mixed strategy by an arbitrarily small amount. In other words, in infinite time the player has played as well as the best pure action (since the strategy minimizing the loss of the row player will be a one-point distribution).

Boosting. Let \mathcal{X} be a finite space of instances, $\mathcal{X} = \{x_1, \dots, x_n\}$, and \mathcal{H} a finite space of hypotheses $h : \mathcal{X} \rightarrow \{0, 1\}$, $|\mathcal{H}| = m$. Let c be an unknown concept, $c : \mathcal{X} \rightarrow \{0, 1\}$. $c(x)$ is called a label of x . Denote:

$$Pr_{x \sim \delta}[A] = \sum_{x \in A} \delta(x) = \sum_{i=1}^n \delta^i \cdot \mathbf{1}_{\{x_i \in A\}},$$

where $x \in \mathcal{X}$, $A \subset \mathcal{X}$ and δ is a probability distribution over \mathcal{X} .

Definition 3 (γ -weak learning algorithm). *For $\gamma > 0$, we say that an algorithm is a γ -weak learning algorithm for (\mathcal{H}, c) if, for any probability distribution δ over the set \mathcal{X} , the algorithm takes as input a set of labeled examples distributed according to δ and outputs a hypothesis $h \in \mathcal{H}$ with error at most $\frac{1}{2} - \gamma$.*

Given a weak learning algorithm WeakLearn, the goal of boosting is to run it many times on many distributions, and to combine the resulting set of hypotheses into a final hypothesis with arbitrarily small error rate. Boosting proceeds in rounds. On each round $t = 1, \dots, T$:

- The booster weights the observations in data and normalizes the weights to construct a distribution δ_t which is passed to WeakLearn;
- WeakLearn produces a hypothesis $h_t \in \mathcal{H}$ with error on the weighted data at most $\frac{1}{2} - \gamma$, i.e. $Pr_{x \sim \delta_t}[h_t(x) \neq c(x)] \leq \frac{1}{2} - \gamma$

After T rounds, the weak hypotheses h_1, \dots, h_T are combined into a final hypothesis h_{fin} . As stated in [4], the important issues that have to be considered when designing a boosting algorithm are (i) how to choose distributions, and (ii) how to combine the h_t ’s into a final hypothesis.

Boosting in a game-theoretic setting. Consider a repeated game where in a sequence of rounds, the row player chooses a distribution over the set of instances \mathcal{X} and the column player responds with a strategy chosen from the hypothesis space \mathcal{H} . The boosting game matrix should be of the form: $\mathbf{M}^{i,t} = 1$ if $h_t(x_i) = c(x_i)$ and $\mathbf{M}^{i,t} = 0$ otherwise.

Indeed, this setting corresponds to the intuitive meaning of the boosting procedure. The environment always plays a single value distribution concentrated in each round t on the t th move by having WeakLearn to construct the hypothesis h_t . This maximizes the loss of the learner, as WeakLearn tries to optimize the performance of h_t on his chosen distribution over the training data. The learner adjusts this distribution to assign higher weights on the wrongly predicted examples in the previous round.

3 H-Boosting Methods

The analysis performed in this Section is an application of the results achieved within a more general framework of [2] to the field of theory of repeated games. Our results concerning adaptive strategies provide theoretical basis for applying them to implementation of boosting procedures.

Definition 4 (Hannan-consistent adaptive strategy). A Hannan-consistent adaptive strategy is an adaptive procedure H which for any matrix \mathbf{M} with rows and entries in $[0, 1]$, and for any sequence of mixed strategies $\mathbf{Q}_1, \dots, \mathbf{Q}_t$ played by the environment, produces a sequence of mixed strategies $\mathbf{P}_1, \dots, \mathbf{P}_t$ that satisfies:

$$\frac{1}{t} \sum_{s=1}^t \mathbf{M}(\mathbf{P}_s, \mathbf{Q}_s) \leq \min_{1 \leq i \leq n} \frac{1}{t} \sum_{s=1}^t \mathbf{M}(i, \mathbf{Q}_s) + \Delta_t, \tag{4}$$

for some $\Delta_t \rightarrow 0$ as $t \rightarrow \infty$.

We consider two important classes of adaptive strategies. The exponential adaptive strategy with parameter η is defined by:

$$\mathbf{P}_t^{\text{Exp}} = \frac{\exp(-\eta \sum_{s=1}^{t-1} \mathbf{M}(i, \mathbf{Q}_s))}{\sum_{k=1}^n \exp(-\eta \sum_{s=1}^{t-1} \mathbf{M}(k, \mathbf{Q}_s))} \tag{5}$$

and the polynomial adaptive strategy with parameter p is defined by:

$$\mathbf{P}_t^{\text{P}} = \frac{\left(\left[\sum_{s=1}^{t-1} (\mathbf{M}(\mathbf{P}_s, \mathbf{Q}_s) - \mathbf{M}(i, \mathbf{Q}_s)) \right]^+ \right)^{p-1}}{\sum_{k=1}^n \left(\left[\sum_{s=1}^{t-1} (\mathbf{M}(\mathbf{P}_s, \mathbf{Q}_s) - \mathbf{M}(k, \mathbf{Q}_s)) \right]^+ \right)^{p-1}} \tag{6}$$

Using the following Fact (for derivation and proof, see [2]) we analyze properties of both classes of strategies.

Fact 1. Let $\mathbf{R}_t^{E^i}$ and $\mathbf{R}_t^{P^i}$ stand for the cumulative regret when the learner chooses to play strategy $\mathbf{P}_t^{E^i}$ and $\mathbf{P}_t^{P^i}$, respectively. The following inequalities hold:

- (i) In the case of the exponential adaptive strategies, $\frac{\mathbf{R}_t^{E^i}}{t} \leq \frac{\ln n}{\eta t} + \frac{\eta}{2}$
- (ii) In the case of the polynomial adaptive strategies, $\frac{\mathbf{R}_t^{P^i}}{t} \leq \sqrt{\frac{(p-1)n^{\frac{2}{p}}}{t}}$

Note that by subtracting from both sides of (4) the loss minimized by the row player by choosing the best strategy we obtain the maximal time average cumulative regret:

$$\frac{1}{t} \sum_{s=1}^t \mathbf{M}(\mathbf{P}_s, \mathbf{Q}_s) - \min_{1 \leq i \leq n} \frac{1}{t} \sum_{s=1}^t \mathbf{M}(i, \mathbf{Q}_s) = \max_{1 \leq i \leq n} \frac{\mathbf{R}_t^i}{t} \tag{7}$$

Thus, for a given strategy H , if all components of its corresponding cumulative regret vector approach zero in the time limit, then H is H-adaptive.

By Fact 1, with time growing to infinity each time-average component of the cumulative vector \mathbf{R}_t^E/t approaches $\frac{\eta}{2}$. For the parameter $\eta = \sqrt{2 \ln n}/t$, as proposed by [2], the value $\frac{\eta}{2}$ approaches zero, yielding a Hannan-consistent adaptive strategy. But in the case of η constant, we cannot conclude Hannan-consistency. We call exponential based adaptive strategies $\frac{\eta}{2}$ -Hannan-consistent instead.

The adaptive strategy computed by Algorithm LW is really an exponential adaptive strategy \mathbf{P}^E . Indeed, the strategy from Eq. (2) can easily be obtained by setting $\eta = -\ln \beta$ in Eq. (5).

What follows from the Fact 1, is Hannan-consistency of polynomial adaptive strategies; as time t grows to infinity, all the n components of the time-average cumulative regret vector \mathbf{R}_t^P/t approach zero. Point (ii) of Fact 1 states that in the case of polynomial procedures:

$$\max_{1 \leq i \leq n} \frac{\mathbf{R}_t^{P^i}}{t} \leq \sqrt{\frac{(p-1)n^{\frac{2}{p}}}{t}}. \tag{8}$$

Different values of parameter p in possible strategies yield different bounds for the time average cumulative regret value. An adaptive procedure proposed by Hart and Mas-Collel [8] (denote it HMC after the authors' names) in the field of game theory has the parameter p set to $p = 2$, thus achieving bound

$$\max_{1 \leq i \leq n} \frac{\mathbf{R}_t^{\text{HMC}^i}}{t} \leq \sqrt{n/t}.$$

In [2] $p = 2 \ln n$ is proposed. Denote corresponding polynomial procedure BL (after the authors Bianchi and Lugosi).

Here $\max_{1 \leq i \leq n} \frac{\mathbf{R}_t^{\text{BL}^i}}{t} \leq \sqrt{\frac{(2 \ln n - 1)n^{\frac{1}{\ln n}}}{t}} = \sqrt{\frac{\epsilon(2 \ln n - 1)}{t}}$, which is a stronger bound than for HMC.

By Eq. (7) and Eq. (8) the following is satisfied:

$$\frac{1}{t} \sum_{s=1}^t \mathbf{M}(\mathbf{P}_s, \mathbf{Q}_s) \leq \min_{1 \leq i \leq n} \frac{1}{t} \sum_{s=1}^t \mathbf{M}(i, \mathbf{Q}_s) + \sqrt{\frac{(p-1)n^{\frac{2}{p}}}{t}}. \tag{9}$$

Denote $\Delta_t^P = \sqrt{\frac{(p-1)n^2}{t}}$, clearly $\Delta_t^P \rightarrow 0$ as $t \rightarrow \infty$. Note Eq. (9) is equivalent to the Eq. (3) given by Theorem 1 for Algorithm LW.

Thus, by (9) and (3), the complexity of Δ_t for the LW, HMC and BL algorithms is equal to $O(\frac{1}{\sqrt{t}})$ (we assume n constant).

Consider boosting in a game-theoretic setting, like in Section 2. Let us now generalize the proposed implementation of the boosting procedure by putting an arbitrary Hannan-consistent algorithm H in place of LW. That is, H will be used to compute δ . Let us call this class of boosting algorithms *H-boosting methods*.

Theorem 2. *Let H be H-boosting procedure run on a set of n labeled examples from the space \mathcal{X} using a γ -weak learning algorithm in T rounds, satisfying:*

$$\Delta_T < \gamma. \tag{10}$$

Then the combined classifier h_{fin} produced by H satisfies $\forall_{x \in \mathcal{X}} h_{fin}(x) = c(x)$.

This Theorem is a generalization of results obtained by Freund and Schapire [4] for AdaBoost.LW for all H-boosting procedures. Theorem 2 implies that by the usage of a Hannan-consistent algorithm in computing distributions and combining h_t 's for sufficiently large t into a final hypothesis by simple majority one can address the two important issues stated as crucial for designing of the boosting procedure (see Section 2).

4 Algorithm and Experimental Results

Within the general class of H-boosting methods two different subclasses can be distinguished, depending on whether the arbitrary Hannan-consistent procedure H is exponential or polynomial. In the first case we should refer to H as *E-adaptive strategy* and in the second case as *P-adaptive strategy*. We define *E-boosting* and *P-boosting* classes of boosting methods accordingly.

A general schema for implementation of a H-boosting algorithm is based on employing an arbitrary H-adaptive strategy in the place of the procedure updating the distribution δ . An E-boosting algorithm is obtained by using an exponential strategy defined by Eq. (5). Different variants of algorithms within this class are obtained by setting the parameter η . Similarly, different versions of P-boosting algorithms depend on the value of parameter p in the polynomial strategy (6).

A new algorithm AdaBoost.P (Algorithm 1), belonging to the P-boosting class of algorithms is implemented based on polynomial strategy BL, defined by setting $p = 2 \ln n$ (see Section 3). The algorithm takes as an input the number T of iterations to execute, the set of training examples and the weak learning algorithm WeakLearn. By Theorem 2 the speed of convergence of a given strategy H determines the lower threshold for the number of iterations T the respective H-boosting algorithm should be run with. Appropriate T should be set to satisfy

inequality (10). Here it is enough if $\sqrt{\frac{e(2 \ln n - 1)}{t}} < \gamma$. That is, AdaBoost.P should be run at least

$$T > \frac{e(2 \ln n - 1)}{\gamma^2} \tag{11}$$

number of times. In this version of the algorithm, as it is in the case of other known boosting algorithms, it is required to set T arbitrarily, since in most cases for a given WeakLearn the value of γ is usually not known.

Algorithm 1: AdaBoost.P

Input: number of iteration steps T ,
 examples $\{(x_1, c(x_1)), \dots, (x_n, c(x_n))\}$,
 γ -weak learning algorithm **WeakLearn**.

Initialize:

$$\forall_{i=1, \dots, n} \mathbf{R}_0^i = 0$$

$$\forall_{i=1, \dots, n} \delta_1^i = \frac{1}{n}$$

for all $t = 1, \dots, T$ **do**

Train **WeakLearn** using δ_t .

Get back a hypothesis $h_t : \mathcal{X} \rightarrow \{-1, 1\}$ such that:

$$Pr_{x \sim \delta_t} [h_t(x) \neq c(x)] \leq \frac{1}{2} - \gamma$$

Update cumulative regret vector

$$\forall_{i=1, \dots, n} \mathbf{R}_{t+1}^i = \mathbf{R}_t^i + \sum_{i=1}^n \delta_t^i \cdot \mathbf{1}_{\{h_t(x_i) \neq c(x_i)\}} - \mathbf{1}_{\{h_t(x_i) = c(x_i)\}}$$

Set the new weights to be

$$\forall_{i=1, \dots, n} \delta_{t+1}^i = \frac{([\mathbf{R}_{t+1}^i]^+)^{(2 \ln n - 1)}}{\sum_{j=1}^n ([\mathbf{R}_{t+1}^j]^+)^{(2 \ln n - 1)}}$$

Output: the final hypothesis $h_{fin} = \text{sign}(\sum_{t=1}^T h_t(x))$.

To make all the examples equally important, their normalized weights given by distribution δ are made uniform in the initial step. Throughout the procedure, a cumulative regret vector is maintained, which is initialized to a zero vector.

In each step t of the iteration, WeakLearn is trained on the δ_t -weighted examples and produces a hypothesis h_t . Each i th component of the cumulative regret vector is updated by adding the difference between the accuracy WeakLearn achieved with respect to δ_t and the accuracy it would have achieved if δ_t was a degenerate distribution localized at example x_i . Note that in this way if h_t

correctly classifies x_i then $\mathbf{1}_{\{h_t(x_i)=c(x_i)\}} = 1$ and the regret is smaller by $1 - \delta_t^i$ than it would be otherwise.

Next, the new distribution is calculated using the polynomial strategy formula (6), ensuring that the more "regretted" (more often misclassified in previous steps) examples get a higher weight.

Finally, all the hypotheses produced in the series of T steps are combined by a majority vote to produce a final hypothesis h_{fin} .

Results. Performance of the new algorithm was verified on benchmark and mass spectrometry data sets.

Benchmark data sets include the Banana (repository of G. Rätsch), Breast-cancer and Waveform data sets (the UCI Repository). Benchmark tests were performed 100 times for the same data set on the different realizations and the results averaged. The weak learning algorithm used was decision stumps. Performance of AdaBoost.P was comparable to the "weighted majority" boosting and the standard AdaBoost. For the Banana data set all the three algorithms achieved average accuracy of c.a. 86%, for BreastCancer c.a. 70%. In the case of the Waveform data set, AdaBoost.P achieved 88%, and the other two 87%.

Table 1. Comparison of classification results for different classifiers (AdaBoost.P, AdaBoost – standard discrete AdaBoost and CART - decision trees) preceded by dimension reduction (PCA) or biomarker selection (PPC - Peak Probability Contrasts [9]) methods. Best results for each combination of the methods are bolded.

| Classification method | Reduction/Selection method | Dimension/ Feature no. | Accuracy | Sensitivity | Specificity |
|-----------------------|----------------------------|------------------------|---------------|---------------|---------------|
| AdaBoost.P | PCA | 10 | 67.03% | 70.21% | 63.63% |
| | | 50 | 65.93% | 61.70% | 70.45% |
| | PPC | 50 | 72.52% | 74.46% | 70.45% |
| 300 | | 65.93% | 68.08% | 63.63% | |
| AdaBoost | PCA | 10 | 62.63% | 65.95% | 59.09% |
| | | 50 | 64.83% | 65.95% | 63.63% |
| | PPC | 50 | 70.32% | 78.72% | 61.36% |
| 300 | | 65.93% | 65.95% | 65.91% | |
| CART | PCA | 10 | 56.04% | 57.44% | 54.54% |
| | | 50 | 61.53% | 61.70% | 61.36% |
| | PPC | 50 | 59.34% | 65.96% | 52.27% |
| 300 | | 57.14% | 57.45% | 56.81% | |

Further tests were performed on ovarian cancer mass spectrometry data set, first provided by Wu et.al. [10], and later analyzed by Tibshirani et.al. [9]. The data set consists of MALDI-MS spectra on pre-treatment serum samples of 89 subjects, consisting of 42 non-cancer controls and 47 ovarian cancer patients. The MS spectra are measured at 91,360 sites, spaced 0.019 Da apart and extending from 800 to 3500 Da.

The size of the data set requires applying dimension reduction or feature selection prior to classification. Since the type and setting of the prior methods significantly change the performance of the latter, we take them into account in our reports. The results, averaged over five times repeated 10-fold cross-validation runs, are presented in Table [11](#).

5 Discussion

The original contribution here was to introduce the H-boosting class of algorithms, and to implement a new boosting algorithm AdaBoost.P.

Our framework provides tools for analysis of all H-boosting procedures. Section [3](#) give a game-theoretic explanation of high performance of boosting as an ensemble of weak classifiers. By analyzing a run of a H-boosting procedure in terms of a repeated play of a certain game, we showed that it tends to minimize the average regret vector. Thus the aim of such a procedure is to converge to the best strategy possible.

An interesting idea would be to provide an adaptive version of AdaBoost.P, where the number of all iterations T would adjust according to error rates of the weak learning algorithm achieved during the run of the boosting procedure described by the formula [\(11\)](#).

References

1. L. Breiman. Prediction games and arcing algorithms, *Neural Comput.*, 11 (7):1493–1517, 1999.
2. N. Cesa-Bianchi and G. Lugosi. Potential-based algorithms in on-line prediction and game theory, In *Machine Learning*, 51:239 – 261, 2003.
3. M. Culp, K. Johnson, and G. Michailidis. ada: an r package for boosting. In *Journal of Statistical Software*, 2005. (submitted).
4. Y. Freund and R. E. Schapire. Game theory, on-line prediction and boosting, In *Computational Learning Theory*, pp 325-332, 1996.
5. J. Hannan. Approximation to Bayes risk in repeated plays, In *Contributions to the Theory of Games*, 3:97–139, 1957.
6. N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Inf. Comput.*, 108(2):212–261, 1994.
7. R Development Core Team. R: A language and environment for statistical computing. *R Foundation for Statistical Computing*, Vienna, Austria, 2004.
8. S. Hart and A. Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. In *Econometrica*, 68(5):1127-1150, 2000.
9. R. Tibshirani *et al.* Sample classification from protein mass spectrometry, by 'peak probability contrasts'. *Bioinformatics*, 20(17):3034-3044, 2004.
10. B. Wu *et al.* Comparison of statistical methods for classification of ovarian cancer using mass spectrometry data, *Bioinformatics*, 19(13):1636-1643, 2003.

Evolutionary Approach to the Game of Checkers

Magdalena Kusiak, Karol Wałędzik, and Jacek Mańdziuk

Faculty of Mathematics and Information Science, Warsaw University of Technology,
Plac Politechniki 1, 00-661 Warsaw, Poland

mandziuk@mini.pw.edu.pl, madziaq@gazeta.pl, karw@gazeta.pl

Abstract. A new method of genetic evolution of linear and nonlinear evaluation functions in the game of checkers is presented. Several practical issues concerning application of genetic algorithms for this task are pointed out and discussed. Experimental results confirm that proposed approach leads to efficient evaluation functions comparable to the ones used in some of commercial applications.

1 Introduction

In our previous work [1] a comparison was made between two evolutionary heuristic generators applied to the game of give-away checkers. In order to check the quality of developed heuristics they were tested against themselves and against TD-GAC program [2,3], which uses temporal difference learning [4].

In this paper the idea of applying evolutionary heuristic generators in two player games domain is evaluated using the game of US checkers. The game is well known and several AI-based approaches to playing checkers has been published, with the most famous ones being Jonathan Schaeffer's Chinook [5] and TDL-Chinook [6]. The paper is particularly motivated by the two "grand Computational Intelligence achievements" - the Samuel's checkers program [7] and Chellapilla and Fogel's *Blondie 24* program [8]. In short the former work empirically proves the efficacy of a combination of self-play and a variant of reinforcement learning as a suitable learning tool for checkers. On the other hand, the other inspiring work - *Blondie 24* - is an apparent example of learning from scratch, without any human guidance, in a knowledge-free regime.

The reminder of the paper is organized as follows: in the next section basic board features used as the components of the evaluation functions are introduced. Section 3 presents different types of both linear and nonlinear heuristical evaluation functions that were generated using the method described in the paper. In Section 4 the evolutionary process is presented in detail together with achieved experimental results. Conclusions and directions for future research are placed in Section 5.

2 Components of the Evaluation Functions

The heuristics described in this paper are relatively simple and made use of some or all of the following parameters, calculated separately for each player:

Eight **simple features**: numbers of (1) pawns and (2) kings; Numbers of safe - i.e. adjacent to the edge of the board - (3) pawns and (4) kings; Numbers of moveable - i.e. able to perform a move other than capturing - (5) pawns and (6) kings (these two last parameters were calculated taking no notice of capturing priority); (7) Aggregated distance of the pawns to promotion line; (8) Number of unoccupied fields on promotion line.

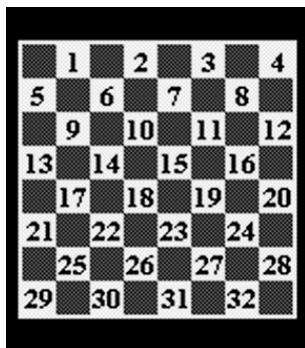


Fig. 1. Notation used for describing patterns. White pawns are at the bottom.

Eleven **layout features**: (9) number of defender pieces - i.e. together pawns and kings situated in the two lowermost rows; (10) Number of attacking pawns - i.e. positioned in three topmost rows; Numbers of centrally positioned - i.e. situated on the eight central squares of the board - (11) pawns and (12) kings; Numbers of (13) pawns and (14) kings positioned on the main diagonal; Numbers of (15) pawns and (16) kings situated on double diagonal; Numbers of loner (17) pawns and (18) kings (a loner piece was defined as the one not adjacent to any other piece); (19) Number of holes - i.e. empty squares adjacent to at least three pieces of the same color.

Six **pattern features** - described below using common notation presented in Fig. 1. All patterns are described from the white player's point of view. Since at most one instance of each pattern can exist for each player in a given board position, features (20)-(25) can take only boolean values. (20) A Triangle - white pawns on squares 27, 31 and 32; (21) An Oreo - white pawns on squares 26, 30 and 31; (22) A Bridge - white pawns on squares 30 and 32; (23) A Dog - white pawn on square 32 and a black one on square 28; (24) A Pawn in the Corner - white man on square 29; (25) A King in the Corner - white king on square 4;

Heuristics could also consider sums of or differences in respective parameters (1)-(25) for both players rather than raw numbers for each player separately.

Two types of heuristics were considered: linear and nonlinear ones. Each linear heuristic consisted of linear combination of the parameters listed above:

$$LinCombOfParam = a_1 \cdot param_1 + a_2 \cdot param_2 + \dots + a_j \cdot param_j, \quad (1)$$

where $1 \leq j \leq 25$ and $param_1, \dots, param_j$ were freely chosen from the above 25 parameters (being either raw numbers or sums or differences calculated for both players).

Nonlinear heuristics were composed of a small number (in our tests 3) of IF-conditions which divided the entire game into disjoint stages (c.f. definitions of heuristics 3Ph and E3Ph in Sect. 3). In each stage the respective linear combination of parameters of the form (II) was considered. As it was the case with linear heuristics sums or differences of particular parameters calculated for both players could be used instead of raw numbers. For either type of heuristics coefficients a_1, \dots, a_j in (II) were optimized in the evolutionary process described in the next section.

3 Types of Heuristics

Initial, simplified tests were carried out in order to find out whether values of respective parameters for both players in linear heuristics exhibited symmetry (i.e. they evolve to approximately opposite values). Pawns counts and kings counts weights were found to be the most asymmetrical. It was therefore obvious that for the symmetrical parameters differences in respective values should be used in order to decrease the number of genes. As for the two asymmetrical parameters, it was suggested that the asymmetry was a result of dependencies between parameters and therefore using differences instead of individual values might affect the quality of heuristics in a negative way. The following linear and nonlinear heuristics were defined.

Linear Heuristics

8 Factors (8F). This heuristic took into consideration differences in values of eight first features ((1) - (8)) listed in sect. 2.

10 Factors (10F). This heuristic took into account the same parameters as 8F, the only exception being the fact that it considered raw numbers of pawns and kings for each player separately (instead of differences), as suggested by the initial symmetry tests described above.

15 Factors (15F). All factors of this heuristic were defined as differences of the respective parameters calculated for both sides. The following features were considered: (1)-(4), (9)-(12), (19)-(25).

19 Factors (19F). All components of this heuristic were in the form of differences between the respective values calculated for both players. It took into account all parameters (1)-(19) described in Sect. 2. It should be noticed that patterns (Dog, Bridge, Oreo, ...) were not included in this heuristics' definition.

25 Factors (25F). This heuristic took into consideration differences in values of all available parameters described in Sect. 2.

Nonlinear Heuristics

The general idea of nonlinear heuristics was based on the fact that it was proved to be advantageous to divide the entire game into several stages and to use different heuristics for different stages. Some crucial moments requiring changing of the evaluation function were identified. These included presence of kings,

which clearly indicated that the game was relatively advanced. End-game positions might also require applying different heuristic. Some tests were carried out which showed that introducing nonlinear components with conditions that were not disjoint hindered the genetic algorithm significantly, which resulted from the fact that having several overlapping conditions made it possible to achieve very similar results in many ways, each time with very different values of parameters. Therefore the game was divided into *disjoint stages*.

3Phase (3Ph). In this heuristic the game was divided into three stages: *Beginning*: each player has more than 3 pawns and no kings are present on the board; *Kings*: both players have more than 3 pieces and at least one king is present; *Ending*: one or both players have 3 pieces or less. Linear heuristics in each stage took into consideration the differences in exactly the same 8 parameters as in 8F heuristics.

Expert 3 Phase (E3Ph). In this heuristic the game was divided into stages in the same way as it was the case of 3Ph. Linear heuristics used in each of the three stages took into consideration differences (white vs. black) in the following ten features : **(3)**, **(10)-(12)**, **(16)**, **(20)-(23)** and **(25)**. The above set of parameters was chosen based on the results obtained in initial runs of the generator – choosing parameters which proved to be the most significant.

In both 3Ph and E3Ph parameters concerning kings were, of course, only considered if kings were present on the board.

4 Evolutionary Process and Results

In order to generate coefficients a_1, \dots, a_j of the above linear combinations genetic algorithms were used. All coefficients of the heuristics were represented as a vector of real numbers whereby each number denoted a single gene (one coefficient). In case of nonlinear heuristics, the conditions that nonlinear components consisted of were not modified by the process of evolution.

Population consisted of 300-400 specimens. In each phase the weakest 75%-80% of the population were regenerated.

Selection – selection was done by means of tournaments. For each tournament a number of specimens were randomly chosen from the population. Their fitness assessments were compared in order to determine the winner of the tournament. Winners of two such tournaments were coupled and went on to crossbreed. Depending on the number of genes of each specimen the size of tournament between specimens was set between two and five.

Crossover – each pair of respective linear combinations in a heuristic (i.e. each nonlinear component and base heuristic function) was crossed over independently. The genotype of each linear combination was randomly partitioned into two. Descendant inherited values of each part of the genotype from respective parent. The value of the gene on which the division was placed was randomly chosen from the interval defined by the values of this gene in parent specimens. The weakest specimen in the population was to be replaced by the newly created descendant, however only if the new specimen's fitness evaluation was greater than that of the one to be replaced.

The ratio of *effective* crossovers (i.e. the ones in which a resulting specimen was actually added to the population) to *potential* crossovers turned out to remain fairly stable throughout the process and ranged between 80% – 90%.

Mutation. Three kinds of mutation could occur in each of the genes of every newly created specimen: multiplying a value of a gene by two, dividing it by two or changing its sign. Multiplying or dividing a value by two were twice as probable as changing the sign.

Heuristic Generator (HG). One of the difficulties encountered while designing a genetic algorithm was defining fitness function for the heuristics. In order to solve this problem the general idea presented in [9] was followed. The game was partitioned into several disjoint stages according to the number of moves already performed. During the first phase of the algorithm a heuristic that would be able to assess correctly situations close to the end of the game was to be obtained. In order to achieve this, alpha-beta algorithm with no heuristic was used to assess a number of randomly generated positions close to the leaves of the game tree. All positions beyond the depth of alpha-beta algorithm were considered a draw. Subsequently, each specimen assessed the same positions and its fitness was calculated according to the formulae $n / \sum (h_i - a_i)^2$, where n denotes the number of test situations, h_i – assessment of the i -th test situation by the heuristic specimen and a_i – by the alpha-beta algorithm.

Once the initial stage had ended, worst fitted fraction of the population was replaced by new random specimens. New board situations closer to the root of the game tree were generated and they were assessed by the alpha-beta algorithm with the fittest specimen of the previous phase used as its heuristic function. The process continued until the root of the game tree was reached. In other words, the evaluation function of HG was evolved step-by-step starting from positions achieved far from the initial position and moving backwards. In each phase a constant fraction of all the test boards came from the stage of the game closest to the beginning (i.e. the stage considered most recently).

In the first step positions obtained in between 82 and 86 moves were considered. This interval was determined based on preliminary tests calculating the average number of moves necessary to finish a game performing random moves. The difference in depths between subsequent phases was set to 6. The number of positions considered in each phase ranged from 3,000 to 4,500.

Heuristics Comparison

In order to determine the relative quality of heuristics a tournament was held, where each heuristic played 10 games with each other with sides swap after each game. Due to some randomness in searching the game tree implemented in alpha-beta, games played between any two heuristics were pairwise different.

Two classifications were used. The first one was based on the total number of games won, tied and lost by each heuristic. The other one took into consideration results of whole 10-game clashes between heuristics. In each case, heuristic would gain 2 points for a win and 1 point for a draw. All the games were played with alpha-beta's search depth limit of 6. The results are presented in Fig. 2.

As might be expected the E3Ph heuristic proved to perform better than any other one, no matter which comparison criteria were considered. More advanced heuristics, considering more sophisticated parameters, also tended to perform well in the tournament, which again was in line with the expectations.

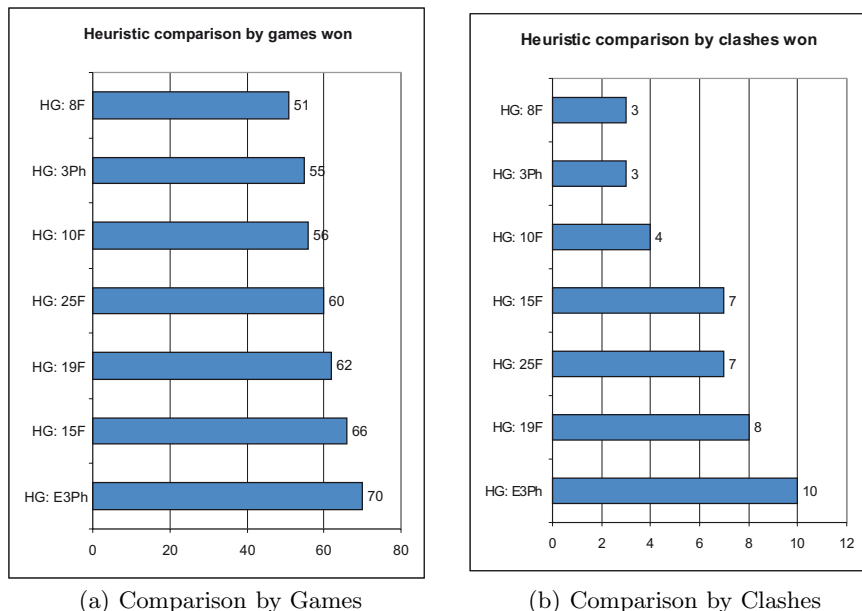


Fig. 2. Heuristic comparison

Comparison with public domain and commercial programs

The piece of research described in this paper was not intended to create a program that could compete against professional or commercial checkers programs. Its main purpose was to assess credibility of evolutionary approach to heuristic generation, in particular with HG method. It was developed with flexibility rather than speed in mind and as a result assessed between 55,000 and 100,000 nodes per second whereas professional checkers applications were sometimes more than one order of magnitude faster. Nevertheless, in order to assess the quality of evolved heuristics, some comparison games were played. In order to make them fair, most of the features of the commercial programs (opening books, endgame databases, hashtables, killer moves identification etc.) had to be disabled or reduced as much as possible. Four best heuristics were chosen for those tests: E3Ph, 15F, 19F and 25F.

The first checkers engine that was tested against the generated heuristics was Simple Checkers (<http://www.fierz.ch/checkers.htm>). Although it is a relatively simple program with publicly available source code, it is at the same time one of the best engines among those not making use of opening books and end-game databases and it surpasses many commercial applications (c.f.

<http://www.bobnewell.net/checkers/checkerprograms.html>). In order to make the comparison fair, Simple Checkers (SC) algorithm was slightly modified so that it would always perform search exactly to the depth of 6 plies, i.e. the same depth as our alpha-beta algorithm. Each heuristic played 50 games against SC. Generally the program appeared to be too strong opponent for developed heuristics. It turned out that the only heuristic capable of winning at least one of 50 games played was E3Ph, achieving a final score of 37 points (1 win, 35 ties, 14 losses). All the other heuristics were capable to draw a few games each: 15F accomplished 9 draws, 25F - 6 draws and 19F - 2 draws.

In the next tests E3Ph heuristic (the undoubtedly best of all heuristics generated) played against shareware version of commercial application Actual Checkers 2000A (<http://www.atlantsoft.com/ach2000a/download.htm>). Due to shareware edition limitation, our heuristic would move first during all of the games (during all other comparisons, sides were swapped after each game). Actual Checkers (AC) program was set to the highest possible difficulty level and was expected to perform a move in 3 seconds on average (which for some reason it wouldn't, taking usually around 5 seconds for each move), which resulted in searches of depth 12 to 18. The number of nodes analyzed by our alpha-beta algorithm was usually well below 1 million per move, which was equivalent to search depth of approximately 9 plies during most of the mid-game but significantly less during end-game. Despite lower search depth E3Ph managed to draw 3 of 4 games played. Single games played with search depths of 9 and 10 ended in draws as well. During the tests it was observed that E3Ph would perform noticeably worse during end-games, often failing to take full advantage of its upper-hand or defend a draw well enough. This probably stemmed from the fact that AC's hashables proved especially useful in this phase of the game.

Finally, in order to once again confirm quality of the E3Ph heuristic, games against another shareware edition of commercial checkers program Mad Checkers (<http://www.sapphiregames.com/madcheckers>) were played. This time alpha-beta's depth limit was set to 8 as usual, but Mad Checkers (MC) engine, being less acclaimed application, was given 0.5s for its move, which was not significantly less than it usually took our program to assess mid-game situations. In order to make the competition even easier for MC, alpha-beta was switched to depth limit of 7 (or in some cases even 6) whenever it happened to perform too slowly during endgames. Nevertheless, our heuristic managed to win 3 out of 4 games with one being a draw because of position repetitions, although MC was left with 3 pieces against our heuristic's 5. In the second 4-game tournament against MC our alpha-beta algorithm was switched to depth limit of 6 plies whilst MC was still admitted 0.5s for a move (i.e. on average more than two times longer than our program would require for a move). This time our heuristic managed to win 1 game and draw 3 others.

5 Conclusions

The focus of this paper is to test usefulness of pure genetic approach for generating evaluation functions in the game of checkers. The underlying assumption

is simplicity and generality of proposed solutions which results in using straightforward genetic operators and shallow game tree search with plain alpha-beta algorithm. In particular, no alpha-beta search enhancements (transposition tables, history heuristics, killer moves, etc.) and no opening books or end-game databases are used.

Not surprisingly, the best of all tested heuristics is E3Ph – a nonlinear 3-phase heuristic. This result confirms the common knowledge in game research that *it is advantageous to divide the entire game into phases* and develop separate heuristics for each part of the game. Specifically it was also observed that game phases need to partition board positions space into *disjoint sets*. Otherwise the genetic process may have difficulties in assigning coefficients for the features shared by two or more game phases.

Another general guideline is *to use differences of respective parameters calculated for both sides* rather than raw numbers separately for both players. This observation was fully confirmed, with the only exception being the case of relatively simple heuristics, where it is recommended that the numbers of pawns and kings be input as raw values (c.f. comparison between 8F and 10F heuristics).

Taking into account the above mentioned assumptions concerning simplicity of proposed approach, the achieved outcomes suggest that usage of genetic algorithms may be a credible way of producing heuristic functions for two-player games. Although heuristics described in this paper did not surpass or even reach the level of play of those created during years of checkers programs development, they are still a challenge for at least some of the commercially available software. This indicates that evolutionary approach, tuned by a careful choice of settings and meta-rules can be successfully applied - particularly to games, for which not enough expert knowledge exists.

References

1. Kusiak, M., Walędzik, K., Mańdziuk, J.: Evolution of heuristics for give-away checkers. In: Proc. ICANN 2005, Part 2, Warszawa, Poland. Volume 3697 of LNCS., Springer-Verlag (2005) 981–987
2. Mańdziuk, J., Osman, D.: Temporal difference approach to playing give-away checkers. In: Proc. ICAISC 2004, Zakopane, Poland. Volume 3070 of LNAI., Springer (2004) 909–914
3. Osman, D., Mańdziuk, J.: Comparison of TDLeaf(λ) and TD(λ) learning in game playing domain. In: Proc. ICONIP 2004, Calcutta, India. Volume 3316 of LNCS., Springer-Verlag (2004) 549–554
4. Sutton, R.: Learning to predict by the method of temporal differences. *Machine Learning* **3** (1988) 9–44
5. Schaeffer, J.: One Jump Ahead: Challenging Human Supremacy in Checkers. Springer-Verlag (1997)
6. Schaeffer, J., Hlynka, M., Jussila, V.: Temporal difference learning applied to a high-performance game-playing program. In: Proc. IJCAI 2001. (2001) 529–534

7. Samuel, A.L.: Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development* **3** (1959) 210–229
8. Fogel, D.B.: *Blondie24: Playing at the Edge of Artificial Intelligence*. Morgan Kaufmann (2001)
9. Borkowski, M.: *Analysis of algorithms for two-player games*. M.Sc. Thesis, Warsaw University of Technology (in Polish) (2000)

Implementation of an Interactive NPC Based on Game Ontology and Game Community Q/A Bulletin Board*

Doo-kyung Park¹, Tae-bok Yoon¹, Kyo-hyun Park¹, Jee-hyong Lee²,
and Keon-myung Lee³

¹ Dept. of Electrical and Computer Engineering, Sungkyunkwan University, Korea
{harderme, tbyoon, megagame}@skku.edu

² Dept. of Electrical and Computer Engineering, Sungkyunkwan University, Korea
jhlee@ece.skku.ac.kr

³ School of Electrical and Computer Engineering, Chungbuk National University, Korea
kmlee@cbnu.ac.kr

Abstract. Recently, there are many researches about NPC (Non-player character) which are appeared in many games. There is a little study which has view of the speech of NPC, but most of them are focused on the movement of NPC. Most of NPC say the same line repeatedly, that make the game unrealistic. In the paper, we construct the quest ontology of game World of Warcraft and collect the articles of Q/A bulletin board which is from the web site, also called community, of the game. With the game ontology and the article from the bulletin, we construct a corpus consisted of game keywords as NPC knowledge. In the runtime, when a player asks a question to NPC, it can find the answer by comparing the question and its own knowledge.

1 Introduction

NPC means the artificial intelligent object in the game. It acts according to a rule programmed by game programmer and it can't be controlled by the game player. Many NPC exist as a component of the background in the game. Some of them deeply related to the game. But most of NPC have little effective to the player. That's why game company doesn't think them important. However, some game developers attempt to give diverse actions to NPC. It makes NPC more intelligent and makes the environment of the game realistic, so it promotes the players' interests. But there still remain unrealistic parts, the messages which are repeated by NPC. So, we make that NPC can speak various and helpful message to the player. Consequently, the player feels the game more realistic and less uncomfortable. To resolve this problem, we propose the way of using Q/A information as a line of NPC it can be collected from the bulletin board of the game community. The system updates NPCs' messages at regular intervals and NPC search the proper message in real-time.

Recently game players don't satisfy only the playing game, but they want share their experiences and information about the game. Game community is the place for sharing their information. Many troubles in a game are solved by Q/A board of the

* This research was supported by the Ubiquitous Computing Technology Research Institute (06B1-B1-10M) funded by Korean Ministry of Science and Technology.

game community [1]. That is, the articles in the game community are good for the basis of the NPCs' line because most of them are related to the game. Above all, it has the answers of the questions that are frequently queried by the players. So, it can be proper candidate message for Q/A situation which can be emerged in the game.

However, there is a trouble to implement an interactive NPC. The game interface is quite different from web site search interface. In the case of web site the people who want some information, they send the query with an index word or sentence. And then they choose the information in the list, which has many results of searching, by themselves. On the other hand, it will be unrealistic that NPC shows every articles as the form of the list which are related to the question, when the player send the query to NPC. Practically, the person, who gets a question from someone, will answer the most proper information by going back over things in his mind.

To find the most exact answer of the question, we must solve that how can we find the best candidates of the similar question which are stored in the Q/A board and how can we choose the best answer which explains the question correctly. In this paper, the special corpuses are constructed by collecting related words of the game, and the relations between of them are defined to provide the ontology information. It is called game ontology and is used to NPC knowledge basis. For the Q/A matching, the query from the player is disjointed by the game ontology and then the most proper answer article is selected by matching between the player's query ontology and NPCs' knowledge. A process of deducing the answer is similar to the question, but we consider special information, the recommendation. This function can be used when the other player, who had same question of the game, checked some article which showed the excellent guidance of the question. The recommended article has the chance of having more correct information of the answer. So we implement that it has priority over everything.

In section 2, the game ontology is presented. Q/A processing is following in section 3, and the experiment to verify the proposed idea is mentioned in section 4. Finally, we will conclude this paper and present future work.

2 Game Ontology

There is an attempt applying the ontology to the game [2]. According to the genre of the game, the ontology can be constructed differently. In this paper, we construct the ontology of the quest information which is the main element of MMORPG (Massively Multiplayer Online Role Playing Game).

2.1 Quest Ontology

In the MMORPG, the game is progressed through the quest. The quest is the special task which is assigned to the players in the game. They can get a useful item, earn the money and make their character stronger by accomplishing the quest. The example of the quest is displayed next.

In the village 'V', NPC, whose name is 'N', ask to the player find the item 'I'. The player goes at the location 'L', and kills the monster, whose name is 'M'. And then the player gets 'I' from M and hand it to 'N', he then compensates with the weapon, 'M'.

Since the quest is composed of various information of the game, if we can classify each of the information according to related quest, it will help to find the related information of the player's query.

2.2 Game Ontology of 'World of Warcraft'

In this paper, we use the quest information of the game 'World of Warcraft (WOW)' which is served by Blizzard [3]. First, we classify the quest information to 3 classes, NPC class, Item class and Quest class. Each class has several properties and they are displayed in table.1 and some of the relations between the classes are explained in table 2.

Table 1. The description of the properties of classes

| Class | Properties | Description |
|--------------|-------------------|--|
| Quest | Name | The name of the quest |
| | Level | The minimum level to perform the quest |
| | Race | The constraint of the races to perform the quest |
| | Task | Kill the enemy, collect the item, send the message |
| | Compensation | Item, money, experience |
| NPC | Name | The name of NPC |
| | Item | The item which can be acquired from the NPC |
| | Location | The location of NPC |
| Item | Name | The name of the item |
| | Acquisition | The location/NPC where the item can be acquired |
| | Performance | The specification of the item |

Table 2. The description of the relations between classes

| Relation | Description |
|-----------------|----------------------------------|
| hasSubquest | This is linked to another quest |
| hasCollection | This is the collection task type |
| hasReward | This is related to the reward |
| hasMonster | This is related to the monster |
| dropItem | This is related to the item |

Fig 1. shows the example, 'Onyxia key quest' in the WOW, It's represented by using quest ontology, which is defined above.

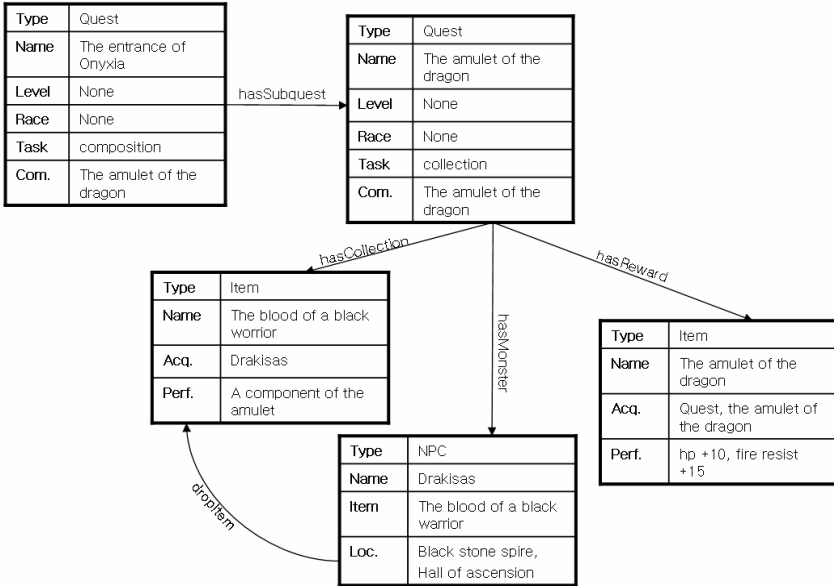


Fig. 1. the representation of the ‘The entrance of Onyxia’ quest ontology information

3 Q/A Using Game Ontology

Because the answer corresponding to the question of the player should be handled as fast as possible, preprocessing must be completed before Q/A matching for finding the answer.

3.1 Preprocessing the Q/A Board in the Game Community

The samples, which are used in this paper, are captured from WOW community. Figure 2 shows the list of articles which are searched from WOW playforum [4].

| | | | | | |
|------|------------------------------------|-------|-------|------|---|
| 8154 | 얼라이언스 오닉입장퀘. 동지입장퀘 [5] | 영성미 | 02/07 | 3659 | 0 |
| 8131 | 공통 오닉시아 막공대 생각하시는분 (공략) | 그림자일격 | 02/01 | 771 | 2 |
| 8088 | 얼라이언스검통과, 오닉 퀘 좀 자세히 가르쳐 주세요.. [2] | 알마도적 | 01/17 | 2288 | 0 |
| 8086 | 공통 오닉머리 먹으면 시작하는퀘 [2] | 카류리오스 | 01/16 | 2455 | 0 |
| 7636 | 얼라이언스 오닉시아열쇠퀘 중에서요... [1] | kensi | 11/05 | 855 | 1 |
| 7381 | 공통 오닉레어퀘 중간에서 끊겼는데 어떻게 .. [2] | 굴단가룻 | 09/26 | 616 | 1 |

Fig. 2. This figure shows the part of the list which displays the result of searching index ‘onix’

The preprocessing is done at regular interval. For the analyzing the article we used Morphological Analyzer developed by natural language processing laboratory in Kookmin university [5]. Each of the questions goes through the necessary formalities.

First of all, the title of the question is analyzed by KMA, and the question title corpus, also called 'keyword' is made by the result of KMA. After processing of the tile, the question content corpus is constructed in the same way. Finally, it finds the most matching quest which has the most plenty keyword of the question.

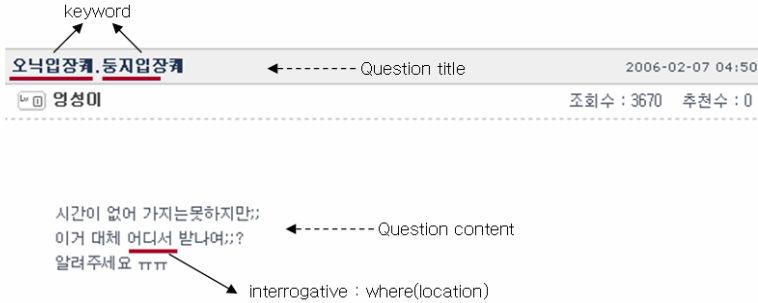


Fig. 3. In the step of morphological analyzing, keywords related to quest ontology and the interrogatives are extracted from the article. In this figure, the word ‘erdiseo’ it means ‘where’ by Korean so ‘erdiseo’ is extracted for the location information to help finding an answer in runtime.

Generally, we can find the information about the name of NPC or the level of the monster from the noun of the question article. And from the interrogative like as ‘where’ or ‘who’, we can get information what kind of the information they need. Using this information NPC finds the article which has the contents related to them. If there are many articles which get a recommendation from other players who have the same question of it, the article which gets the largest votes will be selected. Unless there is recommended one, every answer articles, which are related to the question, will be transformed the answer content corpus by preprocessing using KMA. And the article which has the most plenty keywords or the information matched the question corpus, will be stored as the answer content. Every article in Q/A board of the game community are processed in the same way.

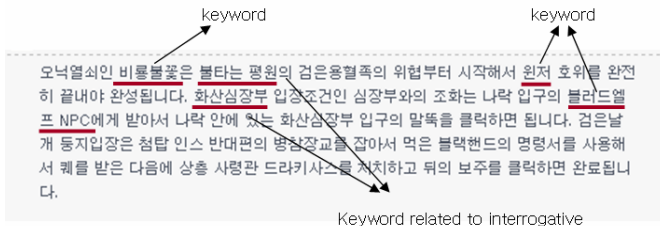


Fig. 4. The processing of answers is similar to the question. Keywords, related to quest ontology and related the interrogatives which are appeared in the question, are extracted from the article. In this figure, ‘the burning plain’ and ‘the heart of the volcano’ are extracted because they are location information.

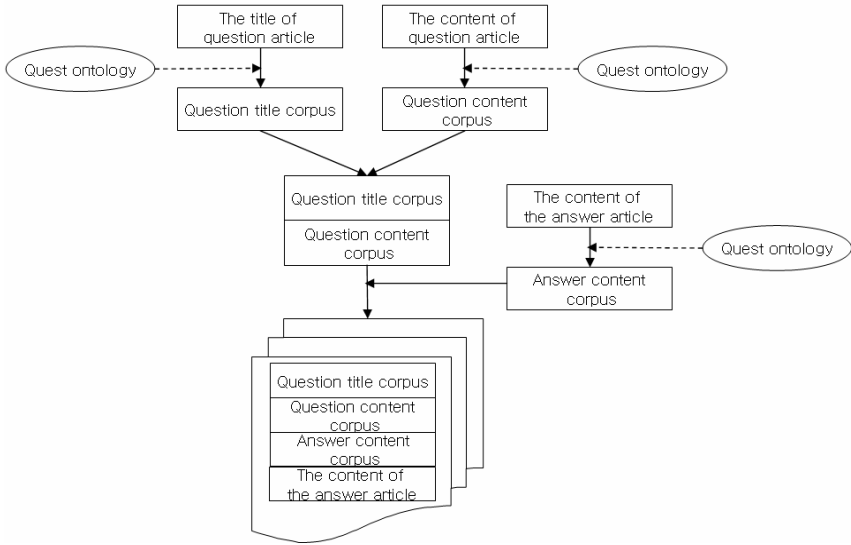


Fig. 5. This figure shows the entire processing which is explained in this chapter. The dotted line means using the quest ontology to extract related keywords. The solid line represents the flow of the process.

3.2 Q/A Processing in the Runtime

When the player asks the question to NPC, the query from the player will be disjointed keywords and interrogatives and make the question corpus by morphological analyzing. Afterward, it's compared with quest ontology and finds the most similar quest then compared with question article corpus in the quest category.

$$\begin{aligned}
 S_{x,y} = & \alpha \left(\frac{\text{the number of keywords of } x \text{ in the question title of } y}{\text{the number of keywords of } x} \right) \\
 & + \beta \left(\frac{\text{the number of keywords of } x \text{ in the question content of } y}{\text{the number of keywords of } x} \right) \\
 & + \gamma \left(\frac{\text{the number of keywords related to interrogatives of } x \text{ in answer content of } y}{\text{the number of keywords in the content of answer of } y} \right)
 \end{aligned}
 \tag{1}$$

x is the query created by the player, y is the question article corpus, and $S_{x,y}$ means the similarity between x and y . α , β , and γ are the weights to control priority of the formula.

The formula (1), which calculates the similarity, is composed of 3 parts, calculating how similar with the title of the question article, the content of the

question article, and corresponding degree between the player question and the data in NPC knowledge. NPC selects the answer which shows the highest similarity in the candidates and send it to the player as the answer message. The candidates are the question article corpuses which show the similarity over the threshold (θ). Unless there are candidates in NPC knowledge, NPC sends the messages that he doesn't have any idea. There are two reasons for existence of the threshold. At first, it must be blocked that NPC says irrelevant story of the question. Second, it's to distribute NPC knowledge for providing more realistic environment to the player. Generally, people have more information about their place of residence but less information other places. So we can give this constraint to NPC through setting the threshold.

4 Experiment and Result

For the simulation of our idea, we collect the articles which are searched by the index 'Onyxia key' in the quest Q/A board in the WOW game community [4]. 34 question articles are collected and 29 of them transformed to question article corpus. 5 of them are discarded because they got no answers. All question articles are classified two classes of the quest. Moreover 78 answer articles, corresponding to 29 questions, are transformed to answer corpus information. And we assumed one of the question content in the question articles as the player question. The details of the selected player question are displayed in figure 6.

| | | |
|-----------------------------|--|--------------------|
| 오닉시아 퀘중에 용의눈케 엔피씨 질문 | | 2005-01-06 07:04 |
| 칼슘사랑 | | 조회수 : 4222 추천수 : 0 |

5명의 폴리모프한 드래곤종 한명이 오닉시아 레어로 갈수있는 열쇠를
만들어 준다가에 침탐 하층에 벨란에게 가보았으나 변화가 없었구요
서부역병 크로미는 안물할페허 여관에 있어서 갔는데 역시나 반응이 없었습니다
남은건이 가시덤불 구름에 벨란이랑 여명의설원 할레
그리고 슬픔의늪의 이타리우스로 알고있는데
혹시나 나머지 3명의 폴리모프 드래곤 위치 아시는분 계시면
답변 좀 부탁드립니다

Fig. 6. This figure shows the selected question article as the player question. The keywords are represented under-bar. But the keywords represented in the content section used as the player question information.

Table 3. This table shows extracted keywords from figure 6 and the result of the classification of the quest

| Type | Keywords |
|------------------------------------|---|
| The content of the player question | 드래곤, 오닉시아(Onyxia), 레어, 열쇠(key), 첨탑(spire), 벨란, 서부역병, 크로미, 안돌할페허, 여관, 가시덤불, 구릉, 벨란, 여명의설원, 할레, 슬픔의눈, 이타리우스, 폴리모프, 위치 |
| The name of the quest | 오닉시아 열쇠 |

Table 3 shows the result of analyzing of the player question using the matching with game ontology.

With the result, it compared the similarity with every answer article in the ‘Onyxia key’ quest. For the convenience we assumed the weights $\alpha = 0.2, \beta = 0.3, \gamma = 0.5$, and the threshold $\theta = 0.3\theta = 0.3$. Table 4 shows the result of the calculation about the candidates which exclude the article which shows the similarity under 0.3.

Table 4. This table informs the result of similarity calculation. NPC will send the message of the answer content of no.1 as a result of the similarity calculation.

| No. | Related keywords | Similarity |
|-----|-----------------------|------------|
| 1 | 할레, 엔피씨, 여명의설원, ... | 0.39 |
| 2 | 벨란, 가시덩쿨, 구릉, 벨란, ... | 0.37 |
| 3 | 할레, 동굴, 마즈소릴, ... | 0.30 |

5 Conclusion and Future Work

It is impossible that NPC communicates with the player completely. But if the game developer constructs the game ontology with effective relation between them and many players share their experiences in the community, we can make NPC more meaning objects in the game through the communication with the player using external knowledge. It helps the player feel the game more realistic and gives more diversity to the game.

There are many challenges to implement our technique in the practical. The game players often use the keyword different way, abbreviated or using a secret language. And the existence of a compound word, which is hard to analyze, drops the performance. To provide clear answer, the filtering of emoticons and slang should be considered.

From now on, we will effort to resolve the trouble mentioned above and refine the similarity formula to find an answer more correctly. Furthermore we will extend the

game ontology beyond the quest Q/A, so it will make NPC send to the player the message that not only the answer of the question but also various type of the stories which are interested in the player.

References

1. Jai Jin Jung, Chung Moo Chang, Tae Ung Kim.: An Exploratory Study for Identifying Key Factors in Online Games Development Strategy Utilizing Web Community. *Journal of Korea Information Processing Society (KISP)*, Vol. 11-D, No. 04, (2004) pp. 991–1002.
2. Yoon Ho-Chang, Heon Hong-Jun, Oh Jung-Suk : A Study on a Proposal of Gaia Game Architecture with Ontology. *Journal of Korea Contents Association KOCON*, Vol. 03, No. 01, (2005) pp. 221–228.
3. World of Warcraft, “<http://www.worldofwarcraft.com/>”
4. World of Warcraft playforum, “<http://www.playforum.net/wow>”
5. Seung-Shik Kang : Korean Morphological Analysis and Information retrieval, Hongpub (2002)

Theory of Saplings Growing Up Algorithm

Ali Karci

Firat University, Department of Computer Engineering,
23119, Elazig, Turkey
akarci@firat.edu.tr

Abstract. The saplings sowing and growing up algorithm (SGA) was inspired by a natural events – evolution of growing up of trees. This algorithm contains two phases: Sowing Phase and Growing up Phase. In this paper, the theoretical foundations of SGA were determined. SGA is defined as a computational model, and it was depicted that there are a collection of Turing Machines for simulating SGA.

1 Introduction

The nature is a field of inspiration for some effective ideas in case of developing and inventing computational methods. In fact, Evolutionary computation is a field of simulating evolution on a computer [1], [2]. There are many computational methods which were inspired by the human beings and natural processes: evolutionary computation [1], [2], [4], [5], [6], [7], artificial immunology [3], etc.

The SGA is fundamentally defined as iterative generation and alternation processes operating on a garden of candidate solutions called saplings [1], [2], [4]-[7]. In general, the SGA process is composed of variation operators generating new candidate solutions with selection limiting the search space. The main variation operators are mating, branching and vaccinating using simple symbolic processing. In addition, the SGA requires neither the differential information nor the continuity involved in the objective function of an optimization problem. Due to this features, SGAs can be easily applied to various optimization problems.

Similar to these methods, Karci et al [10]-[12] introduced a new computational method for searching and optimization problems based on the sowing of saplings, growing up of saplings, and mating of saplings. The sowing of saplings must have equal-length distance in each direction to each other (west, east, north, south), and this is the initial step of method. Then saplings are growing up, and they are mating, branching and there is vaccinating process so, there will be three operators.

SGA consists of two phases: **Sowing Phase**-Uniformed sowing sampling: Saplings are scattered evenly in the feasible solution space. **Growing up Phase**- This phase contains three operators: Mating, branching and vaccinating; details of SGA can be found in [10]-[12].

Eberbach [9] defined the series of Turing Machines for evolutionary computation, and the aim of this paper is to define the Turing Machine model for SGA. We defined SGA as a computational model, and described the collection of Turing Machines for SGA.

The organization of this paper is as follows. In the Sect. 2, we described the cultivating phase and growing up phase of SGA in brief. Section 3 gives the aim of this paper. The experimental results for application of the proposed algorithm to some benchmark functions were given in Sect. 4. Finally, the last section concludes this paper.

2 Saplings Growing Up Algorithm

The evolution of a sapling has two stages: *Cultivating and growing up*.

2.1 Cultivating Saplings

Solution space can be considered as a garden of saplings, and hence all saplings must be scattered in the garden uniformly (Fig.1). If a farmer wants to sow saplings, he will trivially sow them in equal-length distance for the sake of growing up of saplings more quickly (Fig.1). In order to solve a problem by simulating the growing up of saplings, arbitrary solutions to be generated initially must be scattered evenly in the feasible searching space. Each sapling consists of branches, and initially each sapling contains no branches and it is a body. The initial saplings in the garden (initial solution in the search space) can be generated as uniform garden.

Uniform Garden: Suppose that the feasible solution space is Z containing s elements. Then Z can be written as $Z=\{z_0, z_1, \dots, z_{s-1}\}$. We consider the search space as a continuous space and in order to depict the necessity of this method, let us consider binary case. If branch alphabet contains two elements, then it is a binary case and Z is as follows

$$Z=\{00,01,10,11\},$$

and search space size is given by $\text{size}=2^n$ where n is the length of sapling. The set $\{01, 10\}$ is a base of Z , since all remaining elements of Z can be derived from this set by a linear combination of base set elements. This case can be considered as a plane and base set contains unit vector in each dimension.

When the search space has three elements in the base set $\{100, 010, 001\}$, then $s=3^n$. In the continuous case, saplings belong to the space R^3 . Again, remaining elements can be derived from a linear combination of these elements.

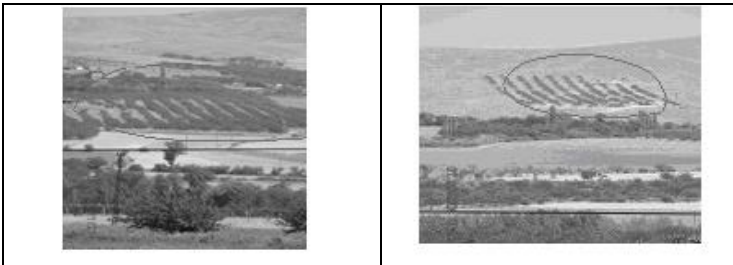


Fig. 1. Scattering saplings in garden uniformly

In general, all vectors in a space can be obtained in a linear combination of elements of base set. If one of elements in the base set is absent, then the dimension corresponding to this element will be vanishing. So, it is important that initial garden must contain saplings which must hold each element of base set. Then cultivating process can get any point in the search space. If saplings do not hold at least one element in the base set, then the point held the absent dimension can be only obtained by branching operation. By considering regularity case and base set, the initial garden must be regular and also hold base set. The obtained garden is called uniform garden.

Initially, two saplings S_0, S_1 are set where $S_0=\{u_1, u_2, \dots, u_n\}$, $S_1=\{l_1, l_2, \dots, l_n\}$, n is the length of sapling and this case is considered as $k=1$ where $u_i, 1 \leq i \leq n$ is the upper bound value for corresponding variable, and $l_i, 1 \leq i \leq n$ is the lower bound value for corresponding variable. Then a dividing factor, let k denote dividing factor, is determined. Firstly, $k=2$ and two extra S_3, S_4 saplings are derived from S_0 and S_1 . The sapling S_0 is divided into two part (equal-length, if possible), in this case, 4 saplings ($2^2=4$) can be derived from S_0 . However, one of them is same as S_0 and another is same as S_1 . Then two saplings which are different from S_0 and S_1 , can be derived:

$$S_3 = \{l_1 + (u_1 - l_1) * r, l_2 + (u_2 - l_2) * r, \dots, l_{n/2} + (u_{n/2} - l_{n/2}) * r, l_{n/2+1} + (u_{n/2+1} - l_{n/2+1}) * (1-r), l_{n/2+2} + (u_{n/2+2} - l_{n/2+2}) * (1-r)\} \text{ and}$$

$$S_4 = \{l_1 + (u_1 - l_1) * (1-r), l_2 + (u_2 - l_2) * (1-r), \dots, l_{n/2} + (u_{n/2} - l_{n/2}) * (1-r), l_{n/2+1} + (u_{n/2+1} - l_{n/2+1}) * r, l_{n/2+2} + (u_{n/2+2} - l_{n/2+2}) * r\},$$

where r is a random number such as $0 \leq r < 1$. The remaining saplings in the garden will be obtained by applying same method with increasing the value of k . This process goes on until the garden is full up. The value of each branch can be binary, and in this case, $S_1 = \{1, 1, \dots, 1\}$ and $S_2 = \{0, 0, \dots, 0\}$. The derived saplings are

$$S_3 = \left\{ \underbrace{1, 1, \dots, 1}_{n/2}, \underbrace{0, 0, \dots, 0}_{n/2} \right\} \text{ and } S_4 = \left\{ \underbrace{0, 0, \dots, 0}_{n/2}, \underbrace{1, 1, \dots, 1}_{n/2} \right\}.$$

In the case of $k=3$, there are 6 derived saplings from S_1 :

$$S_5 = \{l_1 + (u_1 - l_1) * r, l_2 + (u_2 - l_2) * r, \dots, l_{2n/3} + (u_{2n/3} - l_{2n/3}) * r, l_{2n/3+1} + (u_{2n/3+1} - l_{2n/3+1}) * r, \dots, l_n + (u_n - l_n) * r\}$$

$$S_6 = \{l_1 + (u_1 - l_1) * r, l_2 + (u_2 - l_2) * r, \dots, l_{n/3} + (u_{n/3} - l_{n/3}) * r, l_{n/3+1} + (u_{n/3+1} - l_{n/3+1}) * (1-r), \dots, l_{2n/3} + (u_{2n/3} - l_{2n/3}) * (1-r), l_{2n/3+1} + (u_{2n/3+1} - l_{2n/3+1}) * r, \dots, l_n + (u_n - l_n) * r\}$$

$$S_7 = \{l_1 + (u_1 - l_1) * r, l_2 + (u_2 - l_2) * r, \dots, l_{n/3} + (u_{n/3} - l_{n/3}) * r, l_{n/3+1} + (u_{n/3+1} - l_{n/3+1}) * (1-r), \dots, l_n + (u_n - l_n) * (1-r)\}$$

$$S_8 = \{l_1 + (u_1 - l_1) * (1-r), l_2 + (u_2 - l_2) * (1-r), \dots, l_{n/3} + (u_{n/3} - l_{n/3}) * (1-r), l_{n/3+1} + (u_{n/3+1} - l_{n/3+1}) * r, \dots, l_n + (u_n - l_n) * r\}$$

$$S_9 = \{l_1 + (u_1 - l_1) * (1-r), l_2 + (u_2 - l_2) * (1-r), \dots, l_{n/3} + (u_{n/3} - l_{n/3}) * (1-r), l_{n/3+1} + (u_{n/3+1} - l_{n/3+1}) * r, \dots, l_{2n/3} + (u_{2n/3} - l_{2n/3}) * r, l_{2n/3+1} + (u_{2n/3+1} - l_{2n/3+1}) * (1-r), \dots, l_n + (u_n - l_n) * (1-r)\}$$

$$S_{10} = \{ \underbrace{1_1 + (u_1 - l_1) * (1-r)}_{n/3}, \underbrace{l_2 + (u_2 - l_2) * (1-r)}_{n/3}, \dots, \underbrace{l_{2n/3} + (u_{2n/3} - l_{2n/3}) * (1-r)}_{n/3}, \underbrace{l_{2n/3+1} + (u_{2n/3+1} - l_{2n/3+1}) * r}_{n/3}, \dots, \underbrace{l_n + (u_n - l_n) * r}_{n/3} \}$$

In the binary case for k=3,

$$S_5 = \left\{ \underbrace{1, 1, \dots, 1}_{n/3}, \underbrace{1, 1, \dots, 1}_{n/3}, \underbrace{0, 0, \dots, 0}_{n/3} \right\} \text{ and } S_6 = \left\{ \underbrace{1, 1, \dots, 1}_{n/3}, \underbrace{0, 0, \dots, 0}_{n/3}, \underbrace{0, 1, \dots, 1}_{n/3} \right\}.$$

$$S_7 = \left\{ \underbrace{1, 1, \dots, 1}_{n/3}, \underbrace{0, 0, \dots, 0}_{n/3}, \underbrace{0, 0, \dots, 0}_{n/3} \right\} \text{ and } S_8 = \left\{ \underbrace{0, 0, \dots, 0}_{n/3}, \underbrace{1, 1, \dots, 1}_{n/3}, \underbrace{1, 1, \dots, 1}_{n/3} \right\}.$$

$$S_9 = \left\{ \underbrace{0, 0, \dots, 0}_{n/3}, \underbrace{1, 1, \dots, 1}_{n/3}, \underbrace{0, 0, \dots, 0}_{n/3} \right\} \text{ and } S_{10} = \left\{ \underbrace{0, 0, \dots, 0}_{n/3}, \underbrace{0, 0, \dots, 0}_{n/3}, \underbrace{1, 1, \dots, 1}_{n/3} \right\}.$$

2.2 Growing Up Saplings

a) Mating: The aim of mating operator is to generate a new sapling from currently existing saplings by interchanging currently exist information. There will be a mating factor for each pair of saplings, since the distance between a pair is the most important factor which causes the mating of pairs or not.

Let $S_1 = \{s_{1,1}, s_{1,2}, \dots, s_{1,i}, \dots, s_{1,n}\}$ and $S_2 = \{s_{2,1}, s_{2,2}, \dots, s_{2,i}, \dots, s_{2,n}\}$ be two saplings. The distance between S_1 and S_2 affects the mating process' taking place or not, and it depends on the distance between current pair. Let $P_m(S_1, S_2)$ is probability of mating of saplings S_1 and S_2 . Then $P_m(S_1, S_2)$ can be calculated as follow

$$P_m(S_1, S_2) = 1 - \frac{\left(\sum_{i=1}^n (s_{1,i} - s_{2,i})^2 \right)^{1/2}}{R},$$

where $R = \left(\sum_{i=1}^n (u_i - l_i)^2 \right)^{1/2}$, u_i is the upper bound for the corresponding distance between the pair of currently selected saplings, and l_i is the lower bound. The probability of mating of two saplings depends on the distance between both saplings.

b) Branching: In order to grow up a branch on any point on the body of sapling, there will be no near branch previously occurred there. Let $S_i = s_{1,1}s_{1,2} \dots s_{1,i} \dots s_{1,n}$ be a sapling. If a branch occurs in point $s_{1,i}$ (the value of $s_{1,i}$ is changed), then the probability of a branch occurring in point $s_{1,j}$ could be calculated in two ways: linear and non-linear. The distance between $s_{1,i}$ and $s_{1,j}$ can be considered as $|j-i|$ or $l_i - j_l$. If g_i is a branch, then the probability of $s_{1,j}$ being a branch is

$$P(s_{1,j} | s_{1,i}) = 1 - \frac{1}{(j-i)^2}, \quad i \neq j$$

in linear case, and $P(s_{1,j}|s_{1,i})$ is similar to conditional probability, however, it is not pure conditional probability. In the non-linear case, the probability can be considered as

$$P(s_{1,j} | s_{1,i}) = 1 - \frac{1}{e^{(ij-i)^2}}.$$

If $i=j$, then $P(s_{1,j}|s_{1,i})=0$.

c) Vaccinating: The vaccinating process takes place between two different saplings in case of similarity of saplings. Since the similarity of saplings affects the success of vaccinating process, and also vaccinating success is proportional to the similarity of both saplings. In this study, the similarity of saplings is computed in two ways:

$S_1 = \{s_{1,1}, s_{1,2}, \dots, s_{1,i}, \dots, s_{1,n}\}$ and $S_2 = \{s_{2,1}, s_{2,2}, \dots, s_{2,i}, \dots, s_{2,n}\}$ for $1 \leq i \leq n$, $s_{1,i}, s_{2,i} \in \{0,1\}$ $Sim(S_1, S_2) = \sum_{i=1}^n s_{1,i} \oplus s_{2,i}$ The vaccinating process takes place as follow, if

$Sim(S_1, S_2) \geq \text{threshold}$,

$$S'_1 = \begin{cases} s_{1,i} & \text{if } s_{1,i} = s_{2,i} \\ \text{random}(1) & \text{if } s_{1,i} \neq s_{2,i} \end{cases} \quad \text{and} \quad S'_2 = \begin{cases} s_{2,i} & \text{if } s_{2,i} = s_{1,i} \\ \text{random}(1) & \text{if } s_{2,i} \neq s_{1,i} \end{cases},$$

where S'_1 and S'_2 are obtained as consequence of applying vaccinating process to S_1 and S_2 ; $\text{random}(1)$ is a binary number. Saplings are not vaccinated arbitrarily. The saplings to be vaccinated must satisfy the inequality defined on the similarity ($Sim(S_1, S_2) \geq \text{threshold}$). The initial value of threshold depends on the problem solvers. S_1 and S_2 are saplings and the similarity of S_1 and S_2

$$Sim(S_1, S_2) = \sum_{i=1}^n \frac{|s_{1,i} - s_{2,i}|}{u_i - l_i}.$$

If $Sim(S_1, S_2) \geq n * \epsilon$, where ϵ is a user-defined constant ($0 < \epsilon < 1$), then S_1 and S_2 are vaccinated as follows

$$S'_1 = \begin{cases} s_{1,i} & \text{if } \frac{|s_{1,i} - s_{2,i}|}{u_i - l_i} \leq \epsilon \\ s_{2,i} & \text{if } \frac{|s_{1,i} - s_{2,i}|}{u_i - l_i} > \epsilon \end{cases} \quad \text{and} \quad S'_2 = \begin{cases} s_{2,i} & \text{if } \frac{|s_{1,i} - s_{2,i}|}{u_i - l_i} \leq \epsilon \\ s_{1,i} & \text{if } \frac{|s_{1,i} - s_{2,i}|}{u_i - l_i} > \epsilon. \end{cases}$$

2.3 Algorithm

This algorithm applies operator sequentially. The obtained saplings are regarded as temporary solutions. After applications of operators in order mating, branching and vaccinating, the obtained temporary solutions are evaluated. The next generation will be obtained by selecting m best saplings from current generation and temporary solutions.

Algorithm. SGA

$G(t)$ is the garden at generation t .

1. $t \leftarrow 0$ // Initial time
2. SowingSapling($G(t)$) // Initial garden
3. ComputeObjectiveFunction($G(t)$)
4. **While** termination criterion not met **do**
 - 4.1. $G_1(t) \leftarrow$ Mating($G(t)$)
 - 4.2. $G_2(t) \leftarrow$ Branching($G(t)$)
 - 4.3. $G_3(t) \leftarrow$ Vaccinating($G(t)$)
 - 4.4. ComputeObjectiveFunction ($G_1(t) \cup G_2(t) \cup G_3(t)$)
 - 4.5. $G(t+1) \leftarrow$ Selection($G_1(t) \cup G_2(t) \cup G_3(t) \cup G(t)$)
 - 4.6. $t \leftarrow t+1$

3 Turing Machines for SGA

An SGA is a probabilistic search algorithm directed by the objective function maintaining hill climbing at each step. The garden size maintains multiple search points, hill climbing means that only a current search point from the search sapling is remembered, and a termination condition very often is set to the optimum of the objective function. SGA is computational method, and computational model can be defined as follow. Definitions 1, 2, 3 and 4 were adapted from definitions in [9].

Definition 1. Computational method is $CM=(Q, I, O, f)$, where

- | | | |
|-----|---|----------------------------------|
| Q | : | a set of Computational states |
| I | : | the set of inputs for algorithm |
| O | : | the set of outputs for algorithm |
| f | : | computational rules |

Q is the set of subsets of I and O . $f: Q \rightarrow Q$ and it is a unit function for elements of the set O . Assume that the elements of I x_0, x_1, \dots , and

$$x_0 = x \text{ and } x_{k+1} = f(x_k), \quad k \geq 0.$$

If $x_k \in O$ for minimum k value, then computational sequence will terminate. In this case, x_k data is produced from x data.

Definition 2. SGA can be described in the form of the recurrence relation working in a simple iterative loop in discrete time t , called generations, $t = 0, 1, 2, \dots$:

$$G(t + 1) = S(V(B(M(G(t))))),$$

where $G(t)$ is the current garden and $G(t+1)$ is the garden obtained as a consequence of applying Mating, Branching, Vaccinating and Selection operators to $G(t)$. $G(t)$ and $G(t+1)$ are elements of Γ . Γ is the set of all possible gardens with respect to used representation and encoding. S is selection, V is vaccinating, B is branching and M is mating operators.

- $G(0) \in \Gamma$ is the initial garden.
- $G_j \in \Gamma$ is a garden containing at least one optimum or near optimum solution. So, it satisfies the termination condition. The desirable termination condition is the optimum of the objective function $f(G[t])$ of the best sapling in the garden $G[t] \in \Gamma$.

Definition 3. SGA is a computational method $SGA=(Q, I, O, f)$ where Q is the sequence of generations, I is the set of generated gardens, O is the set of gardens. f is the set of rules applied at each generation. So, $f=S(V(B(M(\dots))))$.

The behaviors of SGA were summarized in brief by adapting from the explanation of behaviors of Evolutionary Computation in [9] as follows.

In fact, there is no restriction on the type of representation and encoding used. Encoding depends on the type of problem and modeling manner. The representation is the order of operators to apply saplings in the current garden. Someone may change the order of operators, while he/she apply these operators to garden at the aim of obtaining better sapling(s). Technically, the above means that the domain of the mating operator M , branching operator B , vaccinating operator V , selection operator S , and the objective function f are extended to operate both on the garden under representation G as well as on the encoding of the SGA. The growing up of saplings is not static, it fluctuates, their variation operators (mating, branching, vaccinating) are subject to slow or fast changes, its goal can be modified as well.

The name of this study, SGA, is a traditional behavior such as evolutionary algorithm. Since SGA violates some basic properties of algorithms. Formally, an SGA looking for the optimum of the objective function violates some classical requirements of algorithms. If its termination condition is set to the optimum of the objective function, it may not terminate after a finite number of steps. In order to satisfy the properties of algorithm in this method and SGA remains algorithmic, it has to be stopped after a finite number of generations or when no visible progress is observable.

So, SGA operates on the garden of solutions, and process of variation and selection are probabilistic. However, selection can be handled as deterministically. The objective function is the goal qualified function for solution of problem. Thus such widely understood SGA is a superset of all algorithms, since SGA operates on multiple search points, and many operators are probabilistic. This is consistent with the intuition that adaptive processes are a superset of static processes.

3.1 Turing Machine for SGA

We define a formal model of SGA - a Turing Machine of SGA. The Turing machine for SGA is a collection of Turing Machines working on the garden $G(t)$, $t=0,1,2,\dots$, and the transition function of each machine at each step represents the SGA at corresponding generation ($t=0,1,2,\dots$). This collection of machines (SGA-Ms) can be defined as follows.

Definition 4. SGA-M is a possibly infinite collection of Turing Machines working on the gardens in generations $t=0,1,2, \dots$, where the transition function represents SGA evolving on the current garden $G(t)$ in generations $t=1,2,3, \dots$.

- a) The first garden $G(0)$ is generated by using uniform garden, and this algorithm can be simulated by a Turing Machine $TM(0)$ whose transition function is implement the rules in Algorithm 1. $TM(0)$ generates the $G(0)$.
- b) The output of each generation is a pair $(TM(t+1), G(t+1))$ where $TM(t+1)$ is a Turing Machine and $G(t+1)$ is a garden obtained by applying $f=S(V(B(M(G(t)))))$ for $t=0,1,2,\dots$.
- c) The SGA-M is terminated on $G(t)$ where the current state satisfies the termination condition with the pair $(TM(t), G(t))$ due to the objective function obtaining optimum value or near-optimal value in $G(t)$.
- d) Evaluating objective value for each sapling is $f(TM(t),G(t))=(TM(t), f(G(t)))$ where $f(.)$ is a problem specific objective function and it is independent on the $TM(t)$.

The SGA-M is a collection of Turing Machines where the first TM is different from the other machines in the collection, since the transition function of the first machine represents the algorithm for generating the initial garden. All the remaining TMs are similar, since their transition functions are same which are the applications of Mating, Branching, Vaccinating and Selection on the current garden. The only difference is their gardens.

The operators used in this algorithm are stochastic operators, and the TM whose transition function represents these operators, has more than one tape. There will be at least a random tape or coin-flip state in SGA-M. The weak side of SGA is that it does not remember the history; it just knows the current data. The powerful side of SGA is that it starts at more than one search points.

4 Experimental Results

In order to depicts the superiority of the proposed algorithm, we have used benchmark functions

$$f_1 = \sum_{i=1}^N \left(-x_i \sin \left(\sqrt{|x_i|} \right) \right) \qquad f_2 = \frac{1}{4000} \sum_{i=1}^N x_i^2 - \prod_{i=1}^N \cos \left(\frac{x_i}{\sqrt{i}} \right) + 1$$

$$f_3 = - \sum_{i=1}^N \sin(x_i) \sin^{20} \left(\frac{i x_i^2}{\pi} \right) \qquad f_4 = \frac{1}{N} \sum_{i=1}^N \left(x_i^4 - 16x_i^2 + 5x_i \right)$$

$$f_5 = \sum_{j=1}^{N-1} \left[100(x_j - x_{j+1})^2 + (x_j - 1)^2 \right] \qquad f_6 = \sum_{i=1}^N \left(\sum_{j=1}^i x_j \right)$$

$$f_7 = \max\{|x_i|, i=1,2,\dots,N\}.$$

The problem dimension for $f_1, f_2, f_3, f_4,$ and f_5 is 100 and the problem dimension for remaining benchmark functions is 30 (Table 1 and Table 2). These test functions have many local minimum points that they are challenging enough for performance evaluation, and the existing results reported in [13] can be used for direct comparison.

Table 1. Basic characteristics of the test functions

| Test function | Used feasible solution space | Globally minimal function value (claimed in literature [10]) | Problem dimensions adopted in literature | Number of local minima |
|---------------|------------------------------|--|--|------------------------|
| f_1 | $[-500,500]^N$ | -12569.5 | 30 | NA |
| f_2 | $[-600,600]^N$ | 0 | 30 | NA |
| f_3 | $[0,\pi]^N$ | -99.2784 | 100 | $N!$ |
| f_4 | $[-5,5]^N$ | -78.33236 | 100 | 2^N |
| f_5 | $[-5,10]^N$ | 0 | 100 | NA |
| f_6 | $[-100,100]^N$ | 0 | 30 | NA |
| f_7 | $[-100,100]^N$ | 0 | 30 | NA |

The obtained results in the proposed method show that this method is superior to orthogonal genetic algorithm (Table 2), since it has better results than orthogonal genetic algorithm as seen in Table 2. While the orthogonal genetic algorithm result is -12569.5 for function f_1 , the result obtained in the proposed algorithm is -13520.910 for the same function. While the orthogonal genetic algorithm result is 0.0 for function f_2 , the result obtained in the proposed algorithm is also 0.0 for the same function. While the orthogonal genetic algorithm result is 0.0 for function f_5 , the result obtained in the proposed algorithm is also 0.0 for the same function. While the orthogonal genetic algorithm result is 0.0 for function f_6 , the result obtained in the proposed algorithm is 0.0 for the same function. While the orthogonal genetic algorithm result is 0.0 for function f_7 , the result obtained in the proposed algorithm is 0.0 for the same function.

Table 2. Variable ranges and obtained results in literature and in this study

| Test function | Used feasible solution space | Problem dimensions adopted in this study | Obtained function value by SGA |
|---------------|------------------------------|--|--------------------------------|
| f_1 | $[-500,500]^N$ | 100 | -13520.910 |
| f_4 | $[-600,600]^N$ | 100 | 0 |
| f_7 | $[0,\pi]^N$ | 100 | -98.9057747727068 |
| F_8 | $[-5,5]^N$ | 100 | -77.873208 |
| F_9 | $[-5,10]^N$ | 100 | 0 |
| f_{13} | $[-100,100]^N$ | 30 | 0 |
| f_{14} | $[-100,100]^N$ | 30 | 0 |

5 Conclusions

This paper describes the formal model for SGA in brief. The theory of SGA is not just single Turing Machine, it composes of a collection of Turing Machines. The first

Turing Machine for generating initial garden is different than all others. The remaining have differences in case of inputs, since they take different garden. In order to make SGA be an algorithm, the collection of Turing Machines have to terminate in a finite time. This case maintains by using the maximum number of generations.

The obtained computational method promises the following properties:

- Mating operator is a global search
- Branching operator is a local search operator
- Branching operator is applied to a single sapling, so, the resulting saplings are similar to parent sapling. In this way, branching operator can be used for clustering.
- Vaccinating operator is applied to two dissimilar saplings for the potential of using good information in both saplings.
- This method does not use an extra function or relation for determination of quality of sapling. It uses just objective function.
- If all saplings are used for the solution, they can be used for multi-objective optimization.
- The proposed algorithm does not need to use parameters for evolution of garden. It just takes the number of saplings in the garden, and the number of branches in a sapling.

Genetic algorithm is a global search method; however, proposed method contains both local and global search steps. Genetic algorithm uses fitness function for determination of quality of chromosomes, however, proposed method does not use any extra function for determination of quality of saplings.

Acknowledgements. This study is supported by Scientific and Technological Research Council of Turkey with grant number EEEAG-105E144.

References

1. Alatas, B. Karci, A., Alli, H.: A Novel Approach in Genetic Evolution for Determination of Weight Distribution of Neural Networks. 3rd International Conference on Mathematical & Computational Applications, ICMCA'2002 (2002) 306-314, Konya-Turkey.
2. Goldberg, D.: Genetic Algorithm in Search, Optimization and Machine Learning. Massachusetts, Addison-Wesley Publishing Company Inc. (1989)
3. Dorigo, M., Maziello, V., Colomi, A.: The Ant System: Optimization by a Colony of Cooperating Ants. IEEE Trans. on Systems, Man and Cybernetics B (1996) 26(1) 29-41
4. Karci, A., Arslan, A.: Bidirectional evolutionary heuristic for the minimum vertex-cover problem. Journal of Computers and Electrical Engineerings (2003) vol. 29, pp.111-120
5. Karci, A., Arslan, A.: Uniform Population in Genetic algorithms, I.U. Journal of Electrical & Electronics (2002) vol.2 (2), pp.495-504
6. Karci, A.: Novelty in the Generation of Initial Population for Genetic Algorithms, Kes-2004: 8th International Conference on Knowledge Based Intelligent Information & Engineerings Systems (2004) pp:268-275
7. Gundogan, K.K., Alatas, B., Karci, A.: Mining Classification Rules by Using Genetic Algorithms with Non-random Initial Population and Uniform Operator. Turkish Journal of Electrical Engineering & Computer Sciences (2004) vol.12, pp:43-52.

8. Karci A.: Turing Machine. Turkish Encyclopaedia of Informatics (2006) pp:778-782 Papatya yayıncılık.
9. Eberbach, E.: The Role of Completeness in Convergence of Evolutionary Algorithms. The 2005 IEEE Congress on Evolutionary Computation (2005) vol:2, pp:1706-1713.
10. Karci, A., Alatas, B., Akin, E.: Sapling Growing up Algorithm, (in Turkish), ASYU-2006 Akıllı Sistemlerde Yenilikler ve Uygulamaları Sempozyumu (2006) pp.57-61, Istanbul
11. Karci, A., Alatas, B.: Thinking Capability of Saplings Growing up Algorithm. 7th International Conference on Intelligent Data Engineering and Automated Learning, Burgos/Spain, LNCS Springer (2006)
12. Karci, A., Alatas B.: Evolutionary Algorithms and Simulating Saplings IKECCO-2006- 3rd Conference on Electronics and Computer Engineering (2006) Bishkek / Kyrgyzstan
13. Leung, Y.W., Wang, Y.: An Orthogonal Genetic Algorithm with Quantization for Global Numerical Optimization. Trans. on Evolutionary Computation (2001) vol:5, No.1, pp: 41-53.

Improved Production of Competitive Learning Rules with an Additional Term for Vector Quantization

Enrique Mérida-Casermeiro¹, Domingo López-Rodríguez¹,
Gloria Galán-Marín², and Juan M. Ortiz-de-Lazcano-Lobato³

¹ Department of Applied Mathematics,
University of Málaga, Málaga, Spain
{merida,dlopez}@ctima.uma.es

² Department of Electronics and Electromechanical Engineering,
University of Extremadura, Badajoz, Spain
gloriagm@unex.es

³ Department of Computer Science and Artificial Intelligence,
University of Málaga, Málaga, Spain
jmortiz@lcc.uma.es

Abstract. In this work, a general framework for developing learning rules with an added term (perturbation term) is presented. Many learning rules commonly cited in the specialized literature can be derived from this general framework. This framework allows us to introduce some knowledge about vector quantization (as an optimization problem) in the distortion function in order to derive a new learning rule that uses that information to avoid certain local minima of the distortion function, leading to better performance than classical models. Computational experiments in image compression show that our proposed rule, derived from this general framework, can achieve better results than simple competitive learning and other models, with codebooks of less distortion.

1 Introduction

Vector quantization (VQ) is a coding method designed to represent a multidimensional space by means of a finite number of vectors, called representatives or prototypes. A vector quantizer maps each input vector in the p -dimensional Euclidean space \mathbb{R}^p into one of the K prototypes. The construction of a vector quantizer can be modelled as an optimization problem in which a distortion function is minimized. If the set of input vectors is finite, $X = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, and the set of K prototypes (the codebook) is given by $\{\mathbf{w}_1, \dots, \mathbf{w}_K\}$, an usual measure of the distortion introduced in the coding process is given by:

$$F(\mathbf{W}) = \frac{1}{N} \sum_{i=1}^N \min_{j=1, \dots, K} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \quad (1)$$

where the matrix $\mathbf{W} = (\mathbf{w}_1, \dots, \mathbf{w}_K)$ can also be considered as a vector with Kp components.

Among the most popular applications of VQ one can find image and speech signals compression. VQ can also be considered as an approach to data clustering by means of combinatorial optimization techniques which divide the data into clusters according to a suitable cost (or distortion) function, like the one given in [1].

According to Shannon's rate distortion theory, VQ can always achieve better compression performance than any conventional coding technique based on the encoding of scalar quantities [1].

In its beginnings, the high amount of computation required by existing encoding techniques did not allow the use of VQ techniques. Linde, Buzo and Gray [2] proposed the well-known LBG algorithm for VQ which made no use of differentiation, and it is the standard approach to compute the codebook. While the LBG algorithm converges to a local minimum, it is not guaranteed to reach the global minimum.

Competitive neural networks are designed to cluster the input data. Thus, by using VQ techniques in this type of networks, tasks such as data coding and compression can be performed. This fact explains that the competitive learning is an appropriate algorithm for VQ of unlabelled data. A multitude of VQ techniques were developed in conjunction with competitive networks: Ahalt, Krishnamurthy and Chen [3] developed a training algorithm for designing VQ codebooks with near-optimal results, that can be used to develop adaptive vector quantizers. Yair, Zeger and Gersho [4] proved certain convergence properties of the Kohonen algorithm for VQ design, and also introduced the so-called soft competition scheme, which updates all the codevectors simultaneously with a step size that is proportional to its probability of winning. Pal, Bezdek and Tsao [5] proposed a generalization of learning VQ for clustering which avoids the necessity of defining an update neighbourhood scheme and the final centroids do not seem sensitive to initialization. The rival penalized competitive learning was introduced by Xu, Krzyzak and Oja [6]. In this new algorithm for each input not only the winner unit is modified to adapt itself to the input, but also its rival unlearns with a smaller learning rate. Ueda and Nakano [7] presented a new competitive learning algorithm with a selection mechanism based on the equidistortion principle for designing optimal vector quantizers. The selection mechanism enables the system to escape from local minima. Uchiyama and Arbib [8] showed the relationship between clustering and VQ and presented a competitive learning algorithm which generates units where the density of input vectors is high and showed its efficiency in color image segmentation based on the least sum of squares criterion. Mao and Jain [9] have proposed a self-organizing network for hyperellipsoidal clustering that is applied to texture segmentation problems. More recently, Gómez-Ruiz and Muñoz-Pérez [10,11] presented two new learning rules based on the principle of maximizing the distance between codevectors, introducing the concept of expansive and competitive learning achieving very good results.

We propose a new heuristic strategy to develop learning rules for competitive networks whose main contribution is the inclusion of an additional term in

the distortion function allowing to escape from local minima when suitably defined. New learning rules can be derived from this generalized distortion function and used as weight update schemes for the network, as proved in the following sections.

2 Construction of Learning Rules with Additional Terms

In this section we will suppose that the set of possible solutions $\mathbf{W} = (\mathbf{w}_1, \dots, \mathbf{w}_K)$ to VQ is bounded ($\|\mathbf{W}\| \leq M < \infty$).

We will also consider a sequence of distortion functions $\{F_n\}$ obtained as small perturbations of the original F . The perturbation term decreases as n tends to ∞ and helps the learning process (optimization of the distortion function) to avoid certain local minima, that is, non-global solutions.

The analytic expression for the family of functions F_n considered in this work is:

$$F_n(\mathbf{W}) = \frac{1}{N} \sum_{i=1}^N \left(\min_{j=1, \dots, K} \|\mathbf{x}_i - \mathbf{w}_j\|^2 + \alpha_n \cdot g(\mathbf{x}_i, X, \mathbf{W}) \right) \quad (2)$$

where $g(\mathbf{x}_i, X, \cdot)$ is a differentiable and bounded function (for every i), that is, there exists a number M' such that $\|g(\mathbf{x}_i, X, \cdot)\|_\infty \leq M' < \infty$ and $\{\alpha_n\}$ is a sequence of real numbers converging to 0. This perturbation function g brings all information necessary to avoid local minima, and corresponds to the additional term in the learning rule, as we will see next.

This sequence satisfies one convergence condition (condition of uniform convergence): $\lim_{n \rightarrow \infty} F_n(\mathbf{W}) = F(\mathbf{W})$ for all $\mathbf{W} \in V$, where V is a compact (closed and bounded) subset of \mathbb{R}^{Kp} defined as follows:

$$V = \{\mathbf{W} = (\mathbf{w}_1, \dots, \mathbf{w}_K) = (w_{11}, w_{12}, \dots, w_{1p}, \dots, w_{Kp}) \in \mathbb{R}^{Kp} : \|\mathbf{W}\| \leq M\}$$

This convergence result ensures that the sequence $\{F_n\}$ of “perturbed” distortion functions converges to the original F . This fact implies that the learning rule associated to F can be approximated by the ones associated to the successive F_n , as long as $\lim_{n \rightarrow \infty} \alpha_n = 0$.

To obtain learning rules from the definition of F_n , the stochastic gradient method will be used. This method can be described as follows:

- Consider a random $i \in \{1, \dots, N\}$ and define:

$$T_i(\mathbf{W}) = \min_{j=1, \dots, K} \|\mathbf{x}_i - \mathbf{w}_j\|^2 + \alpha_n \cdot g(\mathbf{x}_i, X, \mathbf{W})$$

- The weight update rule is $\Delta \mathbf{w}_j = -\lambda \frac{\partial T_i}{\partial \mathbf{w}_j} =$

$$= \begin{cases} \lambda(\mathbf{x}_i - \mathbf{w}_j) - \lambda \alpha_n \frac{\partial g(\mathbf{x}_i, X, \mathbf{W})}{\partial \mathbf{w}_j} & \text{if } \mathbf{w}_j = \arg \min_{j=1, \dots, K} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \\ -\lambda \alpha_n \frac{\partial g(\mathbf{x}_i, X, \mathbf{W})}{\partial \mathbf{w}_j} & \text{if } \mathbf{w}_j \neq \arg \min_{j=1, \dots, K} \|\mathbf{x}_i - \mathbf{w}_j\|^2 \end{cases}$$

that is, the winning neuron updates its weight $\mathbf{w} = \mathbf{w}_j$ accordingly to the formula $\lambda(\mathbf{x}_i - \mathbf{w}) - \lambda\alpha_n \frac{\partial g(\mathbf{x}_i, X, \mathbf{W})}{\partial \mathbf{w}}$, that includes the original competitive learning scheme ($\mathbf{x}_i - \mathbf{w}$) plus a perturbation term. For a non-winning neuron, the update is only caused by the perturbation in the distortion function.

The addition of this perturbation term provides a way to include more information in the learning process, as well as a generalization of the usual updating schemes.

By giving different values of the perturbation function g , we can obtain learning rules already known:

1. By defining $g \equiv 0$ (no perturbation), we obtain the classical learning rule:

$$\Delta \mathbf{w}_j = \begin{cases} \lambda(\mathbf{x}_i - \mathbf{w}_j) & \text{if } \mathbf{w}_j = \mathbf{w} \\ 0 & \text{if } \mathbf{w}_j \neq \mathbf{w} \end{cases} \quad (3)$$

2. If we define $g(\mathbf{x}_i, X, \mathbf{W}) = -\|\mathbf{w} - \bar{\mathbf{x}}\|^2$, where \mathbf{w} represents the winning prototype when the input to the net is \mathbf{x}_i , that is,

$$\|\mathbf{x}_i - \mathbf{w}\|^2 = \min_{j=1, \dots, K} \|\mathbf{x}_i - \mathbf{w}_j\|^2 ,$$

we arrive at

$$F_n(\mathbf{W}) = \frac{1}{N} \sum_{i=1}^N (\|\mathbf{x}_i - \mathbf{w}\|^2 - \alpha_n \cdot \|\mathbf{w} - \bar{\mathbf{x}}\|^2) .$$

With this definition, we are trying to minimize the usual distortion $\|\mathbf{x}_i - \mathbf{w}\|^2$ and, at the same time, maximize (note the change of sign) the distance from this prototype to the data centroid, $\|\mathbf{w} - \bar{\mathbf{x}}\|^2$.

As explained before, we can derive a learning rule given by the expression

$$\Delta \mathbf{w}_j = \begin{cases} \lambda \cdot (\mathbf{x}_i - \mathbf{w}) - \lambda \cdot \alpha_n (\bar{\mathbf{x}} - \mathbf{w}) & \text{if } \mathbf{w} = \mathbf{w}_j \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

By naming $\beta_n = \lambda \cdot \alpha_n$, the learning rule described in [10] is obtained.

3. The updating scheme from [11] can be obtained by defining the perturbation term $g(\mathbf{x}_i, X, \mathbf{W}) = \langle \bar{\mathbf{x}}, \mathbf{w} \rangle$ where, as usual, \mathbf{w} is the winning prototype and $\langle \cdot, \cdot \rangle$ is the Euclidean inner product. The associated learning rule is derived:

$$\Delta \mathbf{w}_j = \begin{cases} \lambda \cdot (\mathbf{x}_i - \mathbf{w}) - \lambda \cdot \alpha_n \bar{\mathbf{x}} & \text{if } \mathbf{w} = \mathbf{w}_j \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

To get the same analytic expression as in [11], it suffices to define β_n such that $(1 - \lambda)\beta_n = \lambda\alpha_n$ and substitute in the last expression.

The aim of this rule is to minimize the inner product $\langle \bar{\mathbf{x}}, \mathbf{w} \rangle$, which is achieved when vectors $\bar{\mathbf{x}}$ and \mathbf{w} are in opposite directions, that is $\mathbf{w} \propto -\bar{\mathbf{x}}$.

By defining in an adequate way the perturbation term g , we can obtain multiple learning rules, including most of the mentioned in specialized literature, for example [6] and others. These rules are derived from an optimization problem, where the perturbation term helps to avoid local minima of the original distortion function.

3 A New Learning Rule

As mentioned in [10,11], a pair of necessary and sufficient conditions to ensure that the optimum of F is global are:

- Prototypes (that is, $\mathbf{w}_1, \dots, \mathbf{w}_K$) must be as far away as possible from the centroid of data, that is, the quantity

$$\sum_{j=1}^K n_j \|\mathbf{w}_j - \bar{\mathbf{x}}\|^2 \tag{6}$$

where n_j is the number of data whose associated prototype is \mathbf{w}_j , must be maximized.

- At the same time, prototypes must be the centroids of the set of input patterns represented by them, that is, if

$$S_j = \{\mathbf{x} \in X : \|\mathbf{x} - \mathbf{w}_j\|^2 = \min_{l=1, \dots, K} \|\mathbf{x} - \mathbf{w}_l\|^2\}$$

then it must be satisfied $\mathbf{w}_j = \frac{1}{|S_j|} \sum_{\mathbf{x} \in S_j} \mathbf{x}$.

Our approach is based on these two conditions. The learning rule developed in this paper will try to maximize the value of (6), but in an indirect way.

The two references mentioned earlier [10,11] presented learning rules based on maximize that quantity directly.

We consider an alternative way of maximizing the distance from the prototypes to the data centroid that consists in maximizing the distance from the prototypes to the prototypes centroid and in minimizing the distance between both centroids:

$$\text{maximize } \|\mathbf{w} - \bar{\mathbf{w}}\|^2 \tag{7}$$

$$\text{minimize } \|\bar{\mathbf{x}} - \bar{\mathbf{w}}\|^2 \tag{8}$$

where \mathbf{w} is the winning prototype, and $\bar{\mathbf{w}} = \frac{1}{K} \sum_{j=1}^K \mathbf{w}_j$ is the prototypes centroid.

This new learning rule has an important feature: by (8), the centroid of the prototypes approach the data centroid, so data are better represented by the prototypes. Moreover, since $\bar{\mathbf{w}} \approx \bar{\mathbf{x}}$ in the limit, (7) can be approximately rewritten as maximize $\|\mathbf{w} - \bar{\mathbf{x}}\|^2$. And this implies that the value of (6) is maximized. But, in addition, we have obtained another property of the solution: prototypes centroid is close to data centroid.

Then, the definition of the perturbation term, g , is as follows:

$$g(\mathbf{x}_i, X, \mathbf{W}) = \|\bar{\mathbf{w}} - \bar{\mathbf{x}}\|^2 - \|\mathbf{w} - \bar{\mathbf{w}}\|^2 \tag{9}$$

where \mathbf{w} is the prototype verifying $\|\mathbf{x}_i - \mathbf{w}\|^2 = \min_{j=1, \dots, K} \|\mathbf{x}_i - \mathbf{w}_j\|^2$.

With this definition, the expression for the n -th distortion function F_n is:

$$\begin{aligned}
 F_n(\mathbf{W}) &= \frac{1}{N} \sum_{i=1}^N \left(\min_{j=1, \dots, K} \|\mathbf{x}_i - \mathbf{w}_j\|^2 + \alpha_n \cdot (\|\bar{\mathbf{w}} - \bar{\mathbf{x}}\|^2 - \|\mathbf{w} - \bar{\mathbf{w}}\|^2) \right) \\
 &= F(\mathbf{W}) + \alpha_n \|\bar{\mathbf{w}} - \bar{\mathbf{x}}\|^2 - \frac{\alpha_n}{N} \sum_{j=1}^K n_j \|\mathbf{w}_j - \bar{\mathbf{w}}\|^2
 \end{aligned} \tag{10}$$

This expression shows that by minimizing F_n we are also minimizing F and the expression $\|\bar{\mathbf{w}} - \bar{\mathbf{x}}\|^2$ (that is, $\bar{\mathbf{w}} \approx \bar{\mathbf{x}}$), as well as maximizing the total dispersion of the prototypes $\sum_{j=1}^K n_j \|\mathbf{w}_j - \bar{\mathbf{w}}\|^2$, very related to the maximization of (6), as mentioned before.

The learning rule associated to this F_n is given by:

$$\Delta \mathbf{w}_j = \begin{cases} \lambda(\mathbf{x}_i - \mathbf{w}) + \frac{\lambda \alpha_n}{K}(\bar{\mathbf{x}} - \bar{\mathbf{w}}) - \frac{(K-1)\lambda \alpha_n}{K}(\bar{\mathbf{w}} - \mathbf{w}) & \text{if } \mathbf{w}_j = \mathbf{w} \\ \lambda \frac{\alpha_n}{K}(\bar{\mathbf{x}} - \mathbf{w}) & \text{if } \mathbf{w}_j \neq \mathbf{w} \end{cases} \tag{11}$$

It must be noted that, in this case, non-winning prototypes are also updated, that is, in each step, network weights are completely changed. This fact does not imply an increment of the computational time, since all updates are made in parallel, but it helps to avoid non-optimal solutions.

4 Experimental Results

In this section we illustrate the effectiveness of proposed approach in image compression.

In order to perform image compression with unsupervised learning, the set of input patterns is built by subdividing the gray level image into square subimages named windows. Hence, if the image size is $m \times n$ pixels and the window size is $k \times k$ pixels, we will obtain approximately $\frac{m \times n}{k^2}$ windows. These windows are our input patterns with $p = k \times k$ components (these patterns are obtained by arranging the pixel values row by row from top to bottom). The compression process consists in selecting a reduced set of K representative windows (corresponding to the solution prototypes) and replacing each window of the original image with the closest representative window among the prototypes. In this experiment we have considered a window size of 4×4 pixels and $K = 32$ representative windows. Thus, the neural network has 32 output neurons.

As test images we have used the ones represented in Fig. 1. Each of these images has 256×256 pixels, so the number of input patterns is $N = \frac{256^2}{4^2} = 4096$.

The compression was made by using all of these sequential methods (no batch training is used in this work):

- Simple Competitive Learning (SCL), given by (3).
- Expansive and Competitive Learning from [10] (ECL1), given by (4).



Fig. 1. Test images used in this work: (a) lenna, (b) kids

- Expansive and Competitive Learning from [11] (ECL2), given by (5).
- The proposed algorithm, whose learning rule is described by (11).

After 10 executions of each algorithm, the average distortion over the minimum is showed on Table 1. That is, if for one image the minimum distortion obtained among all the algorithms is m_0 and μ_i represents the average distortion of the i -th algorithm in those 10 executions, the measure of the goodness present in Table 1 is given by:

$$M_i = \frac{\mu_i - m_0}{m_0}$$

Table 1. Average results of the 4 algorithms compared in this work after 10 independent executions

| Image | SCL | ECL1 | ECL2 | Proposed |
|-------|------|-------|------|----------|
| lenna | 2.33 | 22.02 | 3.60 | 0.32 |
| kids | 1.48 | 3.99 | 2.39 | 1.42 |

It can be observed that the proposed algorithm achieves better results on average than the other learning rules compared in this work.

In Fig. 2 we can compare the compression results of the four algorithms on the test images.

If $K = 32$ representatives are used, and window size is 4×4 , then 128 bits are needed to represent each window, but only 5 to represent the code-words, so we may obtain a compression rate of 128 to 5, that is, 25 to 1 approximately.

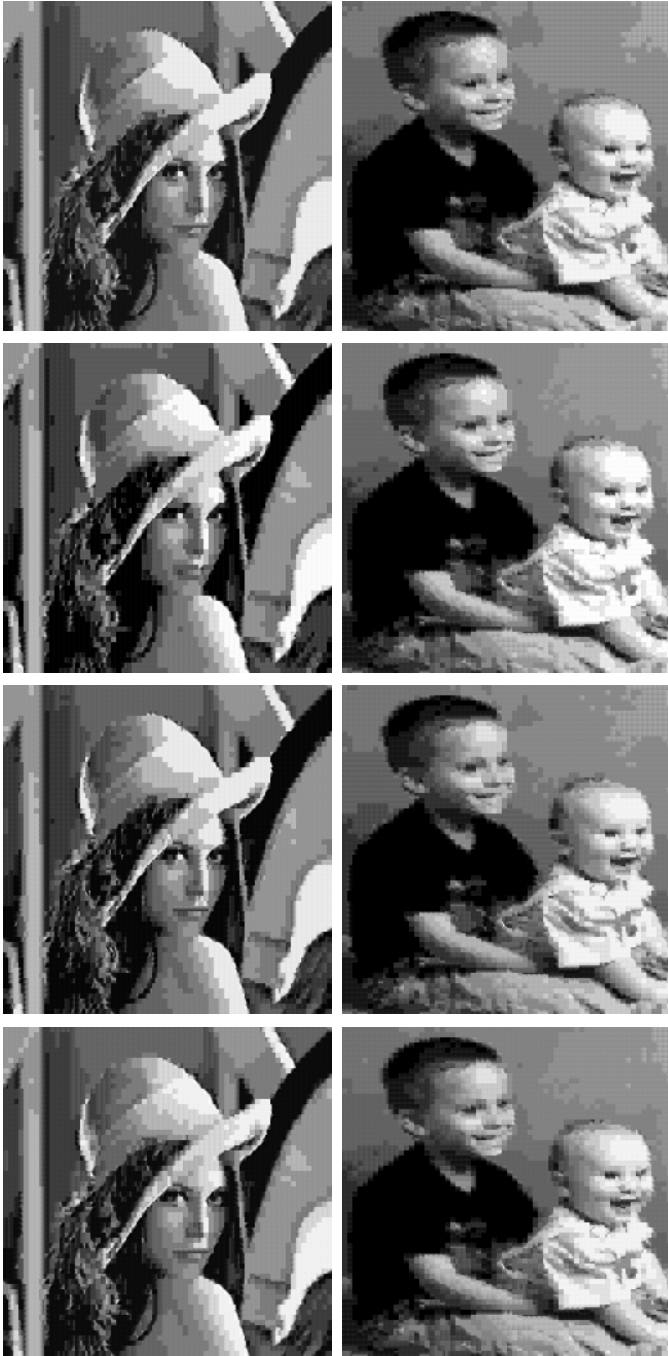


Fig. 2. Compressed images using (from top to bottom): SCL, ECL1, ECL2 and the proposed learning rule

5 Conclusions

In this work we have proposed a general framework for developing learning rules with an added term that plays the role of a perturbation leading to better compression results by including some kind of knowledge about the problem of vector quantization.

This general framework englobes many of the learning rules most commonly cited in the literature, just by defining in a proper way the perturbation term g .

With the help of some previous work [10,11], we have derived a new learning rule that achieves better results by avoiding some local minima of the distortion function, which measures the quality of the compression.

As a future research line, we intend to study the use of frameworks of this kind, generalizing the usual competitive learning, and its convergence to (global or local) minima of the distortion function. It will be interesting to study the convergence in sequential training as well as in batch training.

References

1. Gray, R.M.: Vector quantization. *IEEE ASSP Magazine* **1** (1980) 4–29
2. Linde, Y., Buzo, A., Gray, R.M.: An algorithm for vector quantizer design. *IEEE Trans. on Communications* **28**(1) (1980) 84–95
3. Ahalt, S.C., Krishnamurthy, A.K., Chen, P., Melton, D.E.: Competitive learning algorithms for vector quantization. *Neural Networks* **3** (1990) 277–290
4. Yair, E.K., Zeger, K., Gersho, A.: Competitive learning and soft competition for vector quantizer design. *IEEE Trans. Signal Processing* **40**(2) (1992) 294–308
5. Pal, N.R., Bezdek, J.C., Tsao, E.C.: Generalized clustering networks and kohonens self-organizing scheme. *IEEE Trans. Neural Networks* **4**(4) (1993) 549–557
6. Xu, L., Krzyzak, A., Oja, E.: Rival penalized competitive learning for clustering analysis, rbf net, and curve detection. *IEEE Trans. Neural Networks* **4**(4) (1993) 636–649
7. Ueda, N., Nakano, R.: A new competitive learning approach based on an equidistortion principle for designing optimal vector quantizers. *Neural Networks* **7**(8) (1994) 1211–1227
8. Uchiyama, T., Arbib, M.: Color image segmentation using competitive learning. *IEEE Trans. on Pattern Analysis and Machine Intelligence* **16**(2) (1994) 1197–1206
9. Mao, J., Jain, A.K.: A self-organizing network for hyperellipsoidal clustering (hec). *IEEE Trans. Neural Networks* **7** (1996) 16–29
10. Gómez-Ruiz, J.A., Muñoz Pérez, J., López-Rubio, E., García-Bernal, M.A.: Expansive and competitive neural networks. *Lecture Notes in Computer Science* **2084** (2001) 355 – 362
11. Muñoz Pérez, J., Gómez-Ruiz, J.A., López-Rubio, E., García-Bernal, M.A.: Expansive and competitive learning for vector quantization. *Neural Processing Letters* **15** (2002) 261–273

Reinforcement Learning in Fine Time Discretization

Paweł Wawrzyński

Institute of Control and Computation Engineering,
Warsaw University of Technology,
Nowowiejska 15/19, 00-665 Warsaw, Poland
p.wawrzynski@elka.pw.edu.pl

Abstract. Reinforcement Learning (RL) is analyzed here as a tool for control system optimization. State and action spaces are assumed to be continuous. Time is assumed to be discrete, yet the discretization may be arbitrarily fine. It is shown here that stationary policies, applied by most RL methods, are improper in control applications, since for fine time discretization they can not assure bounded variance of policy gradient estimators. As a remedy to that difficulty, we propose the use of piecewise non-Markov policies. Policies of this type can be optimized by means of most RL algorithms, namely those based on likelihood ratio.

1 Introduction

Reinforcement Learning (RL) algorithms provide solutions to the problem of an intelligent agent that optimizes its behavior in an initially unknown environment. Adaptive control is a very important application of the intelligent agent problem. We would like to construct controllers that are able to “learn” by trial and error to control plants whose dynamics is unknown. The controller may be understood as the agent, and it is rewarded for reaching the control objectives. In present control applications, with fast digital controllers, control stimuli are applied with high frequency. Therefore, each agent’s state results from thousands of previous actions rather than tens like in board games often analyzed as benchmark problems in RL.

Most RL algorithms [1, 9, 4, 5, 7] optimize stationary policies, i.e. ones that draw an action only on the basis of a current state. Application of RL in control systems requires discretization of time. Good control requires fine time discretization. However, a stationary stochastic policy applied to a deterministic system leads to a deterministic behavior of the system for diminishing time discretization [6]. Clearly, this phenomenon precludes exploration capabilities of such policies. Here we analyze the influence of time discretization on variance of policy gradient estimators. In an example we show that stabilization of this variance quickly becomes infeasible as the time discretization decreases.

Our remedy for the fine time discretization problem is based on defining a policy in a special way. Namely, the policy divides agent–environment interaction

into periods such that it relates actions with each others within the same period. Within each period a coherent experiment is carried out that gives a clue to policy improvement. On the basis of a given Markov Decision Process (MDP) and such the policy we define a new MDP and a stationary policy in the new one. We show that each RL algorithm based on likelihood ratio can be applied to optimize the stationary policy in the new MDP.

2 Problem Statement and Likelihood Ratio

We will consider the standard episodic RL setup [8]. A Markov Decision Process is a tuple $\langle \mathcal{S}, \mathcal{A}, P_s, r, P_0, \mathcal{S}^* \rangle$ where \mathcal{S} and \mathcal{A} are the state and action spaces, respectively; $\{P_s(\cdot|s, a) : s \in \mathcal{S}, a \in \mathcal{A}\}$ is a set of state transition probabilities; we write $s_{t+1} \sim P_s(\cdot|s_t, a_t)$. In this work we assume that both \mathcal{S} and \mathcal{A} are multidimensional continuous and each P_s is a density. The immediate reward, r_t depends on the action and the next state, $r_t = r(a_t, s_{t+1})$. P_0 is the distribution of first states of each episode and \mathcal{S}^* is the set of terminal states. The objective of a reinforcement learning is to find a control policy that maximizes future rewards in each state.

We are interested in applications of the solution of the above RL problem to learning control tasks. A painful difficulty that emerges in control problems is the fine time discretization. It makes a single action impact the overall performance insignificantly. Furthermore, the impact of the action emerges a large number (thousands) of steps after the very action took place. We require the learning algorithm work properly no matter how fine the time discretization is.

The problem of Reinforcement Learning is an issue of optimization of a certain performance measure with respect to policy parameters. Because the probability distributions that define the RL problem at hand are unknown, the optimization can not be done directly. A possible approach is to adjust policy parameters along gradient estimators of the performance measure. An important class of such estimators is based on, so called, *likelihood ratio*. Let $f(a; \theta)$ be a density of random variables a of values in \mathcal{A} . f is parametrized by vector $\theta \in \mathfrak{R}^{n_\theta}$. A sample, a , yields a payment, $r(a)$. We are interested in maximization of the expected payment

$$J(\theta) = \mathcal{E}_\theta r(a) = \int_{\mathcal{A}} r(a) f(a; \theta) da.$$

Under certain, quite liberal regularity conditions, for each constant c ,

$$\nabla J(\theta) = \nabla(J(\theta) - c) = \int_{\mathcal{A}} (r(a) - c) \nabla_\theta f(a; \theta) da = \int_{\mathcal{A}} \left((r(a) - c) \frac{\nabla_\theta f(a; \theta)}{f(a; \theta)} \right) f(a; \theta) da$$

and thus

$$(r(a) - c) \frac{\nabla_\theta f(a; \theta)}{f(a; \theta)} = (r(a) - c) \nabla_\theta \ln f(a; \theta)$$

is an unbiased estimator of the gradient $\nabla J(\theta)$. Its variance might be minimized by an appropriate choice of c . The term $\nabla_\theta \ln f(a; \theta)$ is the likelihood ratio.

Reference [2] contains an interesting discussion about the history of its use in RL and other fields.

3 A Stationary Policy for a Continuous-Time System

In this section we analyze by means of a simple example, how the time discretization influences policy gradient estimation. An important insight to this issue has been provided in [6] where it has been shown that in a continuous environment, under quite general conditions, the state trajectory converges to a deterministic limit as the time discretization diminishes. The question arise how this phenomenon influences quality of policy gradient estimators. A general answer to this question is difficult to provide. However, the simple example below suggests that this influence can be demaging.

Let state represent one-dimensional velocity and action represent one-dimensional acceleration. We have $\mathcal{S} = \mathcal{A} = \mathfrak{R}$. An episode lasts for 1 sec. and it includes T steps, $\delta = 1/T$ long each. Within each step an action is drawn from the normal distribution $N(\theta, \sigma_a^2)$ where θ is a policy parameter. The action defines constant acceleration within a step and velocity at the beginning of a trial is null. The only nonzero reward is equal to noised velocity in the last state. We have

$$s_t = \begin{cases} 0 & \text{for } t = 0 \\ s_{t-1} + \delta a_t & \text{for } t > 0, \end{cases} \quad r_t = \begin{cases} 0 & \text{for } t < T - 1 \\ s_T + y_T & \text{for } t = T - 1 \end{cases}$$

where y is a random variable drawn from the normal distribution $N(0, \sigma_y^2)$.

The quality index, $J(\theta)$, of the policy defined by θ is equal to the expected reward at the end of an episode. Because the final reward is a sum of random variables, we have

$$\mathcal{E}_\theta r_{T-1} = \mathcal{E}_\theta \left(\sum_{t=0}^{T-1} \delta a_t + y_T \right) = T\delta\theta = \theta.$$

Therefore, $J(\theta) = \theta$ and $\nabla J(\theta) = 1$. Our main concern here is variance of the policy gradient estimator. We will consider the policy gradient estimator applied in the REINFORCE algorithm [10] since prevailing policy gradient estimators can be considered modifications of this early formula. The estimator applied to our problem is of the form

$$\hat{g} = (r_{T-1} - c) \sum_{t=0}^{T-1} \frac{\partial \ln \pi(a_t; s_t, \theta)}{\partial \theta^r}$$

where c is the *baseline*. Variance of this estimator is defined by the following formula

$$\mathcal{V}\hat{g} = 2 + \frac{1}{\delta\sigma_a^2} ((c - \theta)^2 + \sigma_y^2) \quad (1)$$

derived in the Appendix. We can see that the larger action variance, the smaller gradient estimator variance. The exploration–exploitation balance becomes conspicuous when we compare $\mathcal{V}\hat{g}$ with variance of s_T , namely $\mathcal{V}s_T = \delta\sigma_a^2$ (see the Appendix). By comparing this value with (II), we can see that $\mathcal{V}s_T$ is “almost” inversely proportional to $\mathcal{V}\hat{g}$.

What happens when the time discretization parameter δ decreases? In order to keep gradient estimator variance small, variance of action has to be increased. In fact, variance of gradient estimator remains constant if only

$$\sigma_a^2 \propto 1/\delta.$$

Interestingly enough, this way variance of s_T is also stabilized.

It is seen in our example that in order to keep variance of policy gradient estimator bounded, we have to increase variance of action. However, “actions” in control systems are always bounded; hence, they can not have too large variance either. Therefore, while fine time discretization is necessary for good control it contradicts with quality of policy gradient estimation.

4 MDP Defined by Non-Markov Periods

Let the policy applied by the agent be *piecewise non-Markov* in the following sense. It divides an episode into periods and generates actions within each period on the basis of previous actions and states in this period. Let the periods be indexed by k and k -th period starts at time t_k and lasts for l_k instants. For $i : 0 \leq i < l_k$ we have¹

$$a_{t_k+i} \sim \pi(\cdot; s_{t_k}, a_{t_k}, \dots, s_{t_k+i}, \theta).$$

Let the periods defined by a piecewise non-Markov policy be called *non-Markov periods* or, in short, *nm-periods*.

Given a Markov Decision Process $M = \langle \mathcal{S}, \mathcal{A}, P_s, r, P_0, \mathcal{S}^* \rangle$ we define a new one, $\bar{M} = \langle \bar{\mathcal{S}}, \bar{\mathcal{A}}, \bar{P}_s, \bar{r}, P_0, \mathcal{S}^* \rangle$ with the use of nm-periods defined above. Let states and actions in \bar{M} be denoted by \bar{s} and \bar{a} , respectively, and time be indexed by k . Simultaneously, given a piecewise non-Markov policy π in M , we define a stationary policy $\bar{\pi}$ in \bar{M} generating actions from $\bar{\mathcal{A}}$. States in \bar{M} corresponds to first states in nm-periods; we have

$$\bar{s}_k = s_{t_k} \quad \text{and} \quad \bar{\mathcal{S}} = \mathcal{S}.$$

Actions in \bar{M} corresponds to joint trajectories of states and actions within nm-periods, namely

$$\bar{a}_k = \langle a_{t_k}, s_{t_k+1}, \dots, a_{t_k+l_k-1} \rangle \quad \text{and} \quad \bar{\mathcal{A}} = \bigcup_{i \geq 0} \mathcal{A} \times (\mathcal{S} \times \mathcal{A})^i.$$

¹ We apply “...” also to denote a subsequence of the sequence $(s_1, a_1, s_2, a_2, \dots)$.

The transition distribution in \bar{M} , \bar{P}_s is defined by P_s , π , and the way l_k emerges. In the simplest case $l_k = l$ for a certain constant l unless k -th period is the last one in the episode; then $1 \leq l_k \leq l$. We are free to define the method of calculating rewards in \bar{M} . For instance, a reward in \bar{M} can be an average value of rewards gathered within the corresponding nm-period in M . The distribution of first states P_0 and the set of terminal states \mathcal{S}^* remain unchanged.

An action in \bar{M} is generated by the policy π in tandem with P_s . What is yet important is that we can calculate the likelihood ratio $\nabla_\theta \ln \bar{\pi}(\bar{a}_k; \bar{s}_k, \theta)$. Let us denote

$$S_k = [s_{t_k}^T, \dots, s_{t_k+l_k-1}^T]^T, \quad A_k = [a_{t_k}^T, \dots, a_{t_k+l_k-1}^T]^T, \tag{2}$$

$$\pi_A(A_k; S_k, \theta) = \prod_{i=0}^{l_k-1} \pi(a_{t_k+i}; s_{t_k}, a_{t_k}, \dots, s_{t_k+i}, \theta). \tag{3}$$

π_A is a density of a sequence of actions within an nm-period given a sequence of states. It is entirely defined by the way the policy π generates actions. From the decomposition

$$\bar{\pi}(\bar{a}_k; \bar{s}_k, \theta) = \prod_{i=0}^{l_k-1} \pi(a_{t_k+i}; s_{t_k}, a_{t_k}, \dots, s_{t_k+i} | \theta) \prod_{i=0}^{l_k-2} P_s(s_{t_k+i+1} | s_{t_k+i}, a_{t_k+i}) \tag{4}$$

we see that

$$\nabla_\theta \ln \bar{\pi}(\bar{a}_k; \bar{s}_k, \theta) = \nabla_\theta \ln \pi_A(A_k; S_k, \theta).$$

A special feature of \bar{M} is the fact that the agent is not entirely free to choose an action from $\bar{\mathcal{A}}$. It is hence impossible to apply *Q-Learning* [9] or *SARSA* to \bar{M} . However, optimization of a stationary policy in \bar{M} can be in principle performed by all methods based on likelihood ratio, including episodic *REINFORCE* [10], *Actor-Critics* [4,5,7], *OLPOMDP* [3] and others.

5 Piecewise Non-Markov Policies – Examples

In the present section we define a simple class of non-Markov policies. The policies we suggest exploit each k -th period to carry out a coherent experiment that provides a clue to an improvement of the policy. This coherence is a consequence of the fact that while at each moment the action has a random component, there is a stochastic dependence among these components within the same nm-period. In the next subsection we analyze a way of generating such the stochastically dependent components.

Piecewise independent autoregressive process. Let $\{\epsilon_t, t = 1, 2, \dots\}$ be a sequence of independent random vectors in \mathfrak{R}^n drawn from the normal distribution with zero mean and covariance matrix Σ , i.e. $N(0, \Sigma)$. Also, let $\alpha \in (0, 1)$ and $\{\xi_t, t = 1, 2, \dots\}$ be a sequence of random vectors in \mathfrak{R}^n computed as

$$\xi_t = \begin{cases} \epsilon_t & \text{if } t = t_k \text{ for any } k \\ \alpha \xi_{t-1} + \sqrt{1 - \alpha^2} \epsilon_t & \text{otherwise.} \end{cases} \tag{5}$$

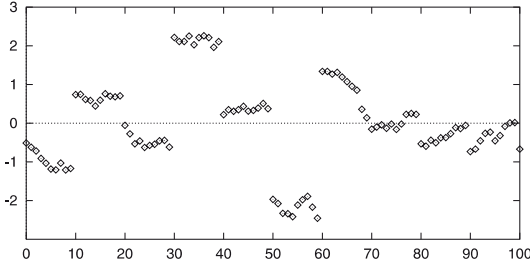


Fig. 1. A run of a piecewise independent autoregressive process for $\Sigma = 1, \alpha = 0.99, t_{k+1} - t_k \equiv 10$. Within an nm-period there is a correlation between random elements while there is no correlation between elements in different nm-periods.

From the above definition it is easy to see, that $\mathcal{E}\xi_t = 0$ for all t . Also, each ξ_t only depends on ϵ -s that belong to the same nm-period. Therefore,

$$\text{cov}(\xi_t, \xi_{t'}) = \mathcal{E}(\xi_t \xi_{t'}^T) = 0$$

for t and t' in different nm-periods. Let us find $\text{cov}(\xi_t, \xi_{t'})$ for t, t' in the same, k -th nm-period. We have

$$\begin{aligned} \xi_t &= \alpha \xi_{t-1} + \sqrt{1 - \alpha^2} \epsilon_t = \alpha^2 \xi_{t-2} + \alpha \sqrt{1 - \alpha^2} \epsilon_{t-1} + \sqrt{1 - \alpha^2} \epsilon_t \\ &= \dots = \alpha^{t-t_k} \epsilon_{t_k} + \sqrt{1 - \alpha^2} \sum_{i=0}^{t-t_k-1} \alpha^i \epsilon_{t-i}. \end{aligned}$$

If $t \leq t'$, then

$$\text{cov}(\xi_t, \xi_{t'}) = \mathcal{E}(\epsilon \epsilon^T) \left(\alpha^{t+t'-2t_k} + (1 - \alpha^2) \sum_{i=0}^{t-t_k-1} \alpha^{2i+t'-t} \right) = \Sigma \alpha^{t'-t}.$$

The result for $t' \leq t$ is symmetrical. Generally, for t, t' in the same nm-period,

$$\text{cov}(\xi_t, \xi_{t'}) = \alpha^{|t'-t|} \Sigma \tag{6}$$

and thus $\{\xi_t, t = t_k, \dots, t_{k+1} - 1\}$ happens to be an autoregressive stochastic process. Notice that $\text{cov}(\xi_t, \xi_t) \equiv \Sigma \equiv \text{cov}(\epsilon_t, \epsilon_t)$.

The random process we defined above, the piecewise independent autoregressive process, may be interpreted as a simple method of transforming normal white noise, ϵ_t , into sequences of random vectors that are stochastically dependent (see Fig. 1). Thank to that dependence, each of these sequences may support a coherent experiment that gives a clue to policy improvement. Below we present two policy that make use of such experiments.

Deterministic Transformation + Stochastic Process. Let an action, a_t , be calculated as

$$a_t = \tilde{a}(s_t; \theta) + \xi_t \tag{7}$$

where $\tilde{a} : \mathcal{S} \times \Theta \mapsto \mathcal{A}$ is a certain deterministic function, e.g. a neural network with input s and weights θ . We will denote by $\nabla\tilde{a}(s, \theta)$ a matrix of derivatives of \tilde{a} with respect to its second argument, namely

$$\nabla\tilde{a}(s, \theta) = \frac{\partial\tilde{a}(s, \theta)}{\partial\theta^T} = \left[\frac{\partial\tilde{a}_j(s, \theta)}{\partial\theta_i} \right]_{i,j}.$$

What we need is to define the distribution $\pi_A(A_k; S_k, \theta)$ and the likelihood ratio $\nabla_\theta \ln \pi_A(A_k; S_k, \theta)$. for S_k and A_k defined in (2). Given trajectory S_k , quantities a_t result from adding random elements ξ_t to constant values $\tilde{a}(s_t; \theta)$. Consequently, the distribution $\pi_A(A_k; S_k, \theta)$ is the normal one with the expected value and the covariance matrix equal to

$$\mathcal{E}A_k = m_k(\theta) = \begin{bmatrix} \tilde{a}(s_{t_k}, \theta) \\ \vdots \\ \tilde{a}(s_{t_k+l_k-1}, \theta) \end{bmatrix}, \quad \text{cov } A_k = C_k = \begin{bmatrix} \Sigma & \alpha\Sigma \cdots \alpha^{l_k-1}\Sigma \\ \alpha\Sigma & \Sigma & & \\ \vdots & & \ddots & \vdots \\ \alpha^{l_k-1}\Sigma & \cdots & & \Sigma \end{bmatrix},$$

respectively. $\text{cov } A_k$ results from the fact that $\text{cov}(a_t, a_{t'}) = \text{cov}(\xi_t, \xi_{t'})$ and (6).

We thus deal with the normal distribution $N(\mathcal{E}A_k, \text{cov } A_k)$ whose mean depends on the parameter θ and variance does not. The density of this distribution is given by

$$\pi_A(A; S_k, \theta) = \left(\sqrt{2\pi}^{l_k n} |C_k| \right)^{-1} \exp \left(-0.5(A - m_k(\theta))^T (C_k)^{-1} (A - m_k(\theta)) \right). \tag{8}$$

We also have

$$\begin{aligned} \nabla_\theta \ln \pi_A(A; S_k, \theta) &= (\nabla_\theta m_k(\theta)) C_k^{-1} (A - m_k(\theta)) \\ &= [\nabla\tilde{a}(s_{t_k}; \theta) \cdots \nabla\tilde{a}(s_{t_k+l_k-1}; \theta)] C_k^{-1} (A - m_k(\theta)). \end{aligned} \tag{9}$$

It seems the most convenient to compute vector $(C_k)^{-1}(A - m_k(\theta))$ as y satisfying the linear equation

$$C_k y = A - m_k(\theta).$$

Deterministic Transformation Of Stochastic Process. Let an action, a_t , be calculated as

$$a_t = \tilde{a}(s_t; \theta + \xi_t). \tag{10}$$

Here, the function $\tilde{a} : \mathcal{S} \times \Theta \mapsto \mathcal{A}$ is defined as previously. However, what is noised here is parameters of \tilde{a} rather than its output. Therefore, the dimension of ξ_t is different than in the previous section. Here $\dim \xi_t = \dim \Theta$ while previously $\dim \xi_t = \dim \mathcal{A}$.

For \tilde{a} smooth with respect to its second argument it is true that

$$\tilde{a}(s_t; \theta + \xi_t) \cong \tilde{a}(s_t; \theta) + \nabla\tilde{a}(s_t, \theta)\xi_t. \tag{11}$$

We will derive $\pi_A(A_k; S_k, \theta)$ and $\nabla_\theta \pi_A(A_k; S_k, \theta)$ assuming that the approximate equation (11) is strict. This assumption is satisfied for \tilde{a} linear in θ . Actions a_t result from an affine transformation of the normal elements ξ_t . Given S_k , the distribution of a_t is also normal with means and covariances equal to

$$\begin{aligned} \mathcal{E}a_t &\cong \tilde{a}(s_t, \theta) + \nabla \tilde{a}(s_t, \theta) \mathcal{E}\xi_t = \tilde{a}(s_t, \theta) \\ \text{cov}(a_t, a_{t'}) &\cong \nabla \tilde{a}(s_t, \theta) \mathcal{E}\xi_t \xi_{t'}^T \nabla \tilde{a}(s_{t'}, \theta)^T = \nabla \tilde{a}(s_t, \theta) \alpha^{|t-t'|} \Sigma \nabla \tilde{a}(s_{t'}; \theta)^T \end{aligned}$$

respectively, for t and t' in the same nm-period. If, additionally, $\Sigma = I\sigma^2$, covariance $\text{cov}(a_t, a_{t'})$ is equal to

$$\text{cov}(a_t, a_{t'}) = \sigma^2 \alpha^{|t-t'|} \nabla \tilde{a}(s_t, \theta) \nabla \tilde{a}(s_{t'}, \theta)^T.$$

The above equation is important because it allows us to avoid computations with Σ which may be a large matrix.

Because A_k is a concatenation of a_t for a sequence of t , the distribution $\pi_A(A_k; S_k, \theta)$ is normal with mean and variance equal to

$$\mathcal{E}A_k = m_k(\theta) = \begin{bmatrix} \tilde{a}(s_{t_k}, \theta) \\ \vdots \\ \tilde{a}(s_{t'_k}, \theta) \end{bmatrix}, \quad \text{cov}A_k = C_k = \begin{bmatrix} \text{cov}(a_{t_k}, a_{t_k}) \cdots \text{cov}(a_{t'_k}, a_{t_k}) \\ \vdots \quad \ddots \quad \vdots \\ \text{cov}(a_{t_k}, a_{t'_k}) \cdots \text{cov}(a_{t'_k}, a_{t'_k}) \end{bmatrix}$$

respectively, where $t'_k = t_k + l_k - 1$. With the above definition of $m_k(\theta)$ and C_k , the density $\pi_A(A; S_k, \theta)$ and the gradient $\nabla_\theta \ln \pi_A(A; S_k, \theta)$ are expressed in (8) and (9), respectively.

6 Discussion

Within the idea presented in this paper a reinforcement learning problem at hand is transformed into the other one and solved by one of the methods based on likelihood ratio. The objective of this transformation is to make RL algorithms better suited to adaptive control problems.

Piecewise non-Markov policies decrease the threat of deterministic behavior of the overall agent-environment system for fine time discretization. For instance, if nm-periods last for $\Delta\tau$ of real time regardless the discretization and there is a strong stochastic dependence between actions within the periods, the thread of deterministicity is headed off and quality of policy gradient estimators is preserved.

The notion of nm-periods introduces a certain additional degree of freedom into a RL process. Each such process includes a sequence of experiments that give clues to policy improvements, usually in the form of policy gradients. Let us consider the question: What should be the length of each such experiment? The answer given by the basic form of Episodic REINFORCE is: the length of an entire episode. Almost all the rest of RL algorithms give the answer that the experiment should last exactly one time step. Within the proposed approach the

answer is between these two extreme possibilities: An experiment lasts for l_k instants where l_k is a controllable parameter.

Third, dividing time into nm-periods enables more flexible treatment the concepts of time and reward. Let us consider algorithms operating on discounted rewards like OLPOMDP or Actor-Critics. The discount factor applied there defines how long, in terms of time, the agent looks ahead optimizing its actions. But it is often natural to look ahead in terms of space rather than time. At present instant t it may be less important what will happen when state becomes far from s_t while it may be quite soon in terms of time. In order to achieve the effect of looking ahead in terms of space, we can define length of a nm-period as

$$l_k = \min\{l : d(s_{t_k}, s_{t_k+l}) > \epsilon\} \quad (12)$$

where d is a certain metric in \mathcal{S} and $\epsilon > 0$ is a threshold. Then, time goes by as fast as state changes. Furthermore, suppose we want to penalize the agent for growing time of accomplishing a certain task. It is possible by assigning a constant penalty to each moment the task is not completed. Apparently, the overall penalty is then proportional to the time of accomplishing the task. But what if we want this penalty to be in a different way related to this time? Within the traditional approach it would be quite problematic. Within our approach, we may define l_k as (12) and introduce the penalty as any function of l_k .

At present we carry out experiments with the proposed methodology. We test it with the use of a simulated 6-degree-of-freedom robotic manipulator. It appears that the performance of existing RL methods in optimization of a stationary control policy for an object of this kind is disappointing. However, when the concept of non-Markov periods is applied, the same methods become surprisingly efficient. We are going to report this work in another paper.

7 Conclusions

We have shown that in RL issues with fine time discretization, keeping variance of policy gradient estimator small may require unfeasibly large action variance. Significance of this difficulty comes from the fact that fine time discretization is typical in control problems. We have proposed a remedy, namely piecewise non-Markov policies. We have shown that combination of existing RL methods with the piecewise non-Markov policies introduces a new degree of freedom into a learning process, namely a length of a sequence of actions that give a clue to policy improvement. Therefore, the piecewise non-Markov policies may be treated on their own right as an enhancement of the existing RL methods.

References

1. A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike Adaptive Elements That Can Learn Difficult Learning Control Problems. *IEEE Transactions on System Man, and Cybernetics*, vol. SMC-13:834-846, 1983.

2. J. Baxter and P. L. Bartlett. Infinite-Horizon Policy-Gradient Estimation, *Journal of Artificial Intelligence Research*, vol. 15:319-350, 2001.
3. J. Baxter, P. L. Bartlett, & L. Weaver. Experiments with Infinite-Horizon, Policy-Gradient Estimation, *Journal of Artificial Intelligence Research*, vol. 15:351-381, 2001.
4. H. Kimura and S. Kobayashi. An Analysis of Actor/Critic Algorithm Using Eligibility Traces: reinforcement learning with imperfect value functions, *Proceedings of the ICML-98*, 1998.
5. V. R. Konda and J. N. Tsitsiklis. Actor-Critic Algorithms. *SIAM Journal on Control and Optimization*, Vol. 42, No. 4:1143-1166, 2003.
6. R. Munos. Policy Gradient in Continuous Time. *Journal of Machine Learning Research* 7, pp. 771-791, 2006.
7. J. Peters, S. Vijayakumar, and S. Schaal. Reinforcement learning for humanoid robotics. *Humanoids2003, 3rd IEEE-RAS International Conference on Humanoid Robots*. Karlsruhe, Germany, Sept.29-30, 2003.
8. R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.
9. C. Watkins and P. Dayan. Q-Learning. *Machine Learning*, vol. 8:279-292, 1992.
10. R. Williams. Simple Statistical Gradient Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, vol. 8:299-256, 1992.

Derivation of Equation 1

π has been defined as the normal distribution $N(\theta, \sigma_a^2)$. Therefore

$$\pi(a_t; s_t, \theta) = \frac{1}{\sqrt{2\pi\sigma_a}} \exp\left(-\frac{1}{2\sigma_a^2}(a_t - \theta)^2\right) \quad \text{and} \quad \frac{\partial \ln \pi(a_t; s_t, \theta)}{\partial \theta^T} = \frac{1}{\sigma_a^2}(a_t - \theta).$$

Consequently

$$\hat{g} = \left(\sum_{t=0}^{T-1} \delta a_t + y - c \right) \left(\sum_{t=0}^{T-1} \frac{1}{\sigma_a^2} (a_t - \theta) \right).$$

Let us define

$$\xi = \sum_{t=0}^{T-1} \delta(a_t - \theta) = s_T - \theta.$$

Obviously

$$\hat{g} = (\xi + y + (\theta - c)) \left(\frac{1}{\delta\sigma_a^2} \xi \right).$$

ξ is a sum of independent random variables. Its distribution is easy to derive as $N(0, \delta\sigma_a^2)$. Note that ξ and s_T have the same variance. Since for each normal random variable X , the equality $\mathcal{E}(X - \mathcal{E}X)^4 = 3(\mathcal{V}X)^2$ holds, we obtain

$$\begin{aligned} \mathcal{V}\hat{g} &= \mathcal{E} \left((\xi + y + (\theta - c))\xi/\delta\sigma_a^2 - 1 \right)^2 \\ &= \mathcal{E} \left(\frac{\xi^4 + 2\xi^3 y + 2\xi^3(\theta - c) + y^2\xi^2 + 2y(\theta - c)\xi^2 + (\theta - c)^2\xi^2}{\delta^2\sigma_a^4} - 2\frac{\xi^2 + y\xi + (\theta - c)\xi}{\delta\sigma_a^2} + 1 \right) \\ &= \frac{3\delta^2\sigma_a^4 + \delta\sigma_a^2\sigma_y^2 + (\theta - c)^2\delta\sigma_a^2}{\delta^2\sigma_a^4} - 2\frac{\delta\sigma_a^2}{\delta\sigma_a^2} + 1 = 2 + \frac{1}{\delta\sigma_a^2}(\sigma_y^2 + (\delta - c)^2). \end{aligned}$$

■

Agent-Based Approach to Solving the Resource Constrained Project Scheduling Problem

Piotr Jędrzejowicz and Ewa Ratajczak-Ropel

Department of Information Systems
Gdynia Maritime University, Poland
{pj,ewra}@am.gdynia.pl

Abstract. JABAT is a middleware supporting the construction of the dedicated A-Team architecture that can be used for solving variety of computationally hard optimization problems. The paper includes a general overview of the JABAT followed by a description and evaluation of the architecture designed by the authors with a view to solving RCPSP and MRCPSPP instances. To construct the proposed system a number of agents, each representing a different optimization algorithm including local search, tabu search, as well as several specialized heuristics have been used. The system has been evaluated experimentally through solving a set of benchmark instances of the RCPSP and MRCPSPP.

1 Introduction

In recent years population based methods as well as agent-based approaches have become successful in solving difficult optimization problems. Among well known population-based methods are, among others, genetic and evolutionary algorithms [6], [15], adaptive memory algorithms [7], ant colony optimization algorithms [5], cultural algorithms [21] as well as hybrid methods [24]. At the same time a number of significant advances have been made in both the design and implementation of autonomous agents. A number of applications of agent technology is growing systematically. Nowadays agent technology is used in the real life supporting successfully variety of industrial and commercial applications. Also a number of agent-based approaches have been proposed to solve different types of optimization problems [1], [17], [4].

One of the successful approaches to agent-based optimization is the concept of an asynchronous team (A-Teams), originally introduced by Talukdar [23]. An A-Team is a collection of software agents that cooperate to solve a problem by dynamically evolving a population of solutions. More precisely, an A-Team is a problem solving architecture where autonomous software agents cooperate to solve problem by sharing access to population of candidate solutions (individuals). The agents works to create, modify or remove individuals from the population. The quality of the solutions evolves as improved solutions are added and poor solutions are removed.

An A-Team usually uses combination of approaches inspired by natural phenomena including, for example, insect societies [16], evolutionary processes [15]

or swarm optimization [13], as well as local search techniques, for example, tabu search [8]. The idea of A-Team has been successfully applied to design various architectures dedicated to solving difficult optimization problems [20], [19].

The paper proposes an agent-based approach to solving instances of the single and multiple mode resource constrained project scheduling problems under the criterion of makespan minimization. RCPSP and MRCPSPP instances are solved using an A-Team environment called JABAT. JABAT is a middleware supporting the construction of the dedicated A-Team architecture that can be used for solving variety of computationally hard optimization problems. The paper includes a general overview of the JABAT followed by a description and evaluation of the architecture designed by the authors with a view to solving RCPSP and MRCPSPP instances. To construct the proposed system a number of agents, each representing a different optimization algorithm including local search, tabu search, as well as several specialized heuristics have been used.

The discussed problems are computationally difficult and belong to the NP-hard class. Because RCPSP and MRCPSPP are important in practical applications many exact and heuristic algorithms have been proposed for solving them (see the reviews [9], [10]).

The following sections of the paper include a short overview of the JABAT environment, RCPSP and MRCPSPP problem formulation, short overview of the functionality and structure of JABAT, a description of the proposed JABAT architecture with details of its implementation, as well as the results of computational experiment carried out to validate the approach. Conclusions focus on evaluation of the system and on directions of future research.

2 JABAT Environment

JADE-based A-Team environment (in short: JABAT) is a middleware supporting the construction of the dedicated A-Team architectures used for solving a variety of computationally hard optimization problems. JABAT has been developed by a team of researchers with the participation of both authors. JABAT engine is JADE, which, in turn, is an enabling technology for the development and run-time execution of peer-to-peer applications which are based on the agents paradigm and which can seamlessly work and interoperate both in wired and wireless environment [2]. From the functional point of view, JADE provides the basic services necessary to distributed peer-to-peer applications in the fixed and mobile environment. JADE allows each agent to dynamically discover other agents and to communicate with team according to the peer-to-peer paradigm. To construct JABAT the Java technologies including the Java 2 Platform Standard Edition (J2SE) with Java Runtime Environment (JRE) have been used.

The problem-solving paradigm on which the JABAT is based can be best defined as the population based approach. The environment is expected to be able to produce solutions to difficult optimization problems through applying the following general rules:

- To solve difficult optimization problems use a set of agents, each representing an improvement algorithm.
- To escape getting trapped into a local optimum generate or construct an initial population of solutions called individuals, which, during computations will be improved by agents, thus increasing chances for reaching a global optimum.

Agent-based architecture of the JABAT allowed implementation of the following features:

- The system can in parallel solve instance of several different optimization problems.
- The optimization processes can be performed on many computers. The user can easily add or delete a computer from the system. In both cases JABAT will adopt to the changes, commanding the agents working within the system to migrate.

The JABAT produces solutions for combinatorial optimization problems using a set of agents. Each agent represents an improvement algorithm. Main functionality of the environment includes organizing and conducting the process of search.

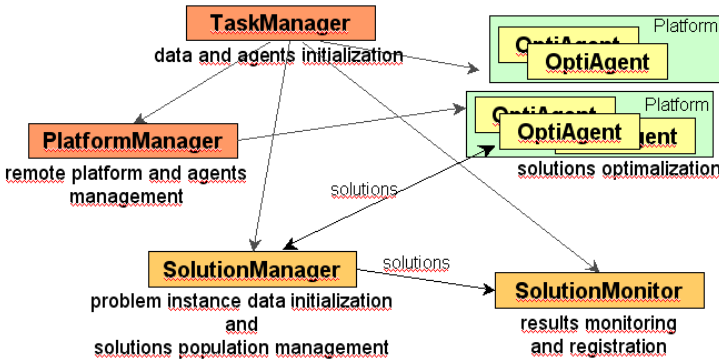


Fig. 1. JABAT general structure

JABAT environment is based on two main agents and three types of special agents. The general structure of the JABAT is shown in Fig. 1. All agents are implemented as Java classes. The main agents are *TaskManager* and *PlatformManger* which manage all other agents and hardware platforms. Both main agents are running continuously on the main platform placed on a server. The special agent types are *SolutionManager*, *SolutionMonitor* and *OptiAgent*. The *OptiAgent* represents the optimization algorithm which is used to solve some particular optimization problem. The *OptiAgent* class must be overwritten by the code specifically designed for solving a particular problem type. The *SolutionManager* manages the population of solutions. Its class may be overwritten

to implement a specific user designed strategy with respect to handling a population of solutions. Such a strategy defines which solutions and when are deleted from the common memory, how they are replaced and how and when new individuals are generated and incorporated into the common memory. The *Solution-Monitor* is responsible to register solutions obtained by *OptiAgents*. The class may be overwritten to make possible recording partial results of computations. One *SolutionManager*, one *SolutionMonitor* and a number, fixed or variable, of *OptiAgents* are run for each problem instance. The detailed description of the JABAT environment may be found in [12].

Apart from the above described JABAT-specific classes several general classes describing a particular optimization problem need to be defined. They include *Data*, *Task*, *Solution*, *DataOntology*, *TaskOntology* and *SolutionOntology*. These classes must be overwritten by a code specific for the considered problem.

3 Resource Constrained Project Scheduling Problem Formulation

A single-mode resource-constrained project scheduling problem consists of a set of n activities, where each activity has to be processed without interruption to complete the project. The dummy activities 1 and n represent the beginning and the end of the project. The duration of an activity j , $j = 1, \dots, n$ is denoted by d_j where $d_1 = d_n = 0$. There are r renewable resource types. The availability of each resource type k in each time period is r_k units, $k = 1, \dots, r$. Each activity j requires r_{jk} units of resource k during each period of its duration where $r_{1k} = r_{nk} = 0$, $k = 1, \dots, r$. All parameters are non-negative integers. There are precedence relations of the finish-start type with a zero parameter value (i.e. $FS = 0$) defined between the activities. In other words activity i precedes activity j if j cannot start until i has been completed. The structure of a project can be represented by an activity-on-node network $G = (SV, SA)$, where SV is the set of activities and SA is the set of precedence relationships. SS_j (SP_j) is the set of successors (predecessors) of activity j , $j = 1, \dots, n$. It is further assumed that $1 \in SP_j$, $j = 2, \dots, n$, and $n \in SS_j$, $j = 1, \dots, n - 1$. The objective is to find a schedule S of activities starting times $[s_1, \dots, s_n]$, where $s_1 = 0$ and resource constraints are satisfied, such that the schedule duration $T(S) = s_n$ is minimized. The above formulated problem is a generalization of the classical job shop scheduling problem and belongs to the class of NP-hard optimization problems [3].

In case of the MRCPSp each activity j , $j = 1, \dots, n$ may be executed in one out of M_j modes. The activities may not be preempted and a mode once selected may not change, i.e., a job j once started in mode m has to be completed in mode m without interruption. Performing job j in mode m takes d_{jm} periods and is supported by a set R of renewable and a set N of non-renewable resources.

The objective is to find a makespan minimal schedule that meets the constraints imposed by the precedence relations and the limited resource

availabilities. It is obvious that the multimode problem can not be computationally easier than the RCPSP.

4 JABAT Architecture for Solving RCPSP and MRCPSPP Instances

In the RCPSP an individual is a schedule represented as a vector of activities $S = [a_1, \dots, a_n]$, each activity a_j is an object consisting of: starting time - s_j , duration - d_j , set of required units of resources, set of predecessors - SP_j and set of successors - SS_j . An individual is represented as an ordered activity list, in which for each activity, all its predecessors are placed at earlier positions and all its successors at later positions on the list. The list serves as a starting point for generating a solution using heuristic known as the serial SGS (Schedule Generation Scheme). Value of the goal function $T(S)$ is directly used as a measure of quality of individuals and hence as a selection criterion when deciding on removing worse individuals from the common memory.

The proposed JABAT architecture for solving RCPSP and MRCPSPP instances includes all types of the required agents such as: *TaskManager*, *Platform-Manager*, *SolutionManager*, *SolutionMonitor* and *OptiAgents*. Agents cooperate together to find the best solution for instances of RCPSP or MRCPSPP. The two most important features affecting the effectiveness of the JABAT architecture are optimization algorithms used by the optimization agents and the strategy according to which the population of solutions evolves. In the proposed system the following algorithms have been implemented as optimization agents:

- Local Search Heuristic (LSA) [11]
- Crossing Heuristic (CH)
- Precedence Tree Heuristic (PTH) [11]
- Tabu Search Algorithm (TSA)
- MCS-based Heuristic (MCSH)

LSA is a simple local search algorithm which finds the local optimum by moving each activity to all possible places in the solution. In the case of MRCPSPP the algorithm additionally checks all possible modes of the activity. CH is based on the one point crossover operator. Two initial solution are crossed until the better solution will be found or all crossing points will be checked. In case of MRCPSPP the algorithm additionally checks all possible modes of the activity. PTH is based on the precedence tree approach proposed in [22]. It finds an optimum solution by enumeration for a partition of the schedule consisting of some activities. Next, it finds the solutions of the successive partitions shifted for a fixed step. The best solution found is remembered. TSA is a tabu search algorithm where the neighborhood of the initial solution is searched by performing moves that are not tabu. In considered TSA the move rely on two activities exchange, in the case of MRCPSPP the move includes modes exchange too. The selected moves are remembered on tabu list. The best solution found is remembered. MCSH is based on the approach proposed in [14]. The possible constraints are detected

based on minimal critical sets (MCS) and adapting shaving method. Next, the solution according to the constraints is created. The procedure is repeated by the fixed number of iteration. Each of the above described algorithm has been implemented as the respective *OptiAgent* class.

The proposed JABAT architecture includes also an implementation of the classes representing the RCPSP and MRCPSP instances. *Data*, *Activity*, *Mode*, *Resource* and *RCPSP_Task* classes have been, respectively, implemented. The *RCPSP_Task* is inherited from the general *Task* class available in JABAT. The *RCPSP_Task* identifies instances which attributes include a list of activities, and a list of available renewable and non-renewable resources. The *Resource* class identifies both - renewable and non-renewable resources, storing the value representing a number of the resource units. The *Mode* class identifies activity modes, which attributes include the mode number, duration and a list of the required resources of both types. Finally, the *Activity* identifies activity, which attributes include the activity number, a list of modes and a list of predecessors and successors. *RCPSP_Solution* class, inherited from the general *Solution* class, has been implemented to describe solutions.

In JABAT, to assure proper communication between agents, ontologies describing considered instances and solutions need to be defined. The *RCPSP_TaskOntology* and *RCPSP_SolutionOntology* classes have been defined through overwriting the *TaskOntology* and *SolutionOntology* classes provided in JABAT.

5 Computational Experiment and Results

To validate the proposed approach computational experiment has been carried out using 2040 benchmark instances of single-mode RCPSP and 3290 instances of the multi-mode MRCPSP. The benchmark data set together with known optimal solution values/upper bounds can be found at [\[18\]](#).

The experiment involved solving all the instances for population consisting of 50 solutions. Initial population was generated randomly. The computation for one problem instance was interrupted after 5 minutes. The results were evaluated in terms of mean and maximum relative error (Mean RE, Max RE) calculated as the deviation from the optimal or best known solutions, percent of solutions equal to the optimal or best known solutions (Exact) and mean computation time (Mean CT) during computation on 6 computers with processor 1.7 GHz. The obtained results are presented in Tables 1-2.

Table 1. Experiment results, single-mode RCPSP

| Number of activities | Mean RE | Max RE | Exact | Mean CT [s] |
|----------------------|---------|---------|-------|-------------|
| 30 | 0.39 % | 7.20 % | 411 | 97 |
| 60 | 0.71 % | 8.85 % | 363 | 109 |
| 90 | 0.85 % | 10.02 % | 336 | 98 |
| 120 | 1.45 % | 10.55 % | 321 | 122 |

Table 2. Experiment results, multiple-mode RCPSP

| Number of activities | Mean RE | Max RE | Exact | Mean CT [s] |
|----------------------|---------|---------|-------|-------------|
| 10 | 0.72 % | 13.16 % | 464 | 46 |
| 12 | 0.73 % | 11.36 % | 455 | 50 |
| 14 | 0.79 % | 11.44 % | 453 | 54 |
| 16 | 0.81 % | 14.14 % | 446 | 55 |
| 18 | 0.95 % | 13.34 % | 450 | 58 |
| 20 | 1.80 % | 13.29 % | 434 | 64 |

6 Conclusions

The proposed agent-based approach to solving instances of the single and multiple mode RCPSP produces good or very good solutions which are competitive or even outperform other population-based approaches. However, main advantages of the developed JABAT architecture which can be directly attributed to the multiple agent system paradigm are:

- Scalability
- Increased effectiveness through mobility and agent migration
- Better use of distributed resources
- Web-based accessibility

Future research will focus on assessing the extend of potential gains in terms of the system effectiveness through increasing a number of optimization agents and computation platforms. Also other improvement algorithms and techniques will be tested with a view to finding best combination of agents to deal with particular project scheduling problem types.

References

1. Aydin, M.E., Fogarty, T.C.: Teams of autonomous agents for job-shop scheduling problems: An Experimental Study. *Journal of Intelligent Manufacturing* **15** (4) (2004) 455–462
2. Bellifemine, F., Caire, G., Poggi, A., Rimassa, G.: JADE. A White Paper, Exp. **3** (3) (2003) 6–20.
3. Blazewicz, J., Lenstra, J., Rinnooy, A.: Scheduling subject to resource constraints: Classification and complexity. *Discrete Applied Mathematics* **5** (1983) 11–24
4. Chang-Shing Lee, Chen-Yu Pan, An intelligent fuzzy agent for meeting scheduling decision support system. Elsevier, *Fuzzy Sets and Systems* **142** (2004) 467–488
5. Dorigo, M., Di Caro, G.: The Ant Colony Optimization Meta-Heuristic, in: D. Corne, M. Dorigo and F. Glover (eds.), *New Ideas in Optimization*, McGraw-Hill, New York (1999) 11–32
6. Goldberg, D.E.: *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, Mass (1989)

7. Golden, B.L., Laptore, G., Taillard, E.D.: An adaptive memory heuristic for class of vehicle routing problems with minmax objective. *Computers and Operations Research*, no. **24** (1997) 445–452
8. Glover, F.: Tabu Search. Part I and II, *ORSA Journal of Computing* **1** (3) (Summer 1990) and **2** (1) (Winter 1990)
9. Hartmann, S., Kolisch, R.: Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem. *European Journal of Operational Research* **127** (2000) 394–407
10. Hartmann, S., Kolisch, R.: Experimental Investigation of Heuristics for Resource-Constrained Project Scheduling: An Update. *European Journal of Operational Research* **174** (2006) 23–37
11. Jedrzejowicz, P., Ratajczak, E.: Population Learning Algorithm for Resource-Constrained Project Scheduling. D.W. Pearson, N.C. Steele, R.F. Albrecht (red.), *Artificial Neural Nets and Genetic Algorithms*, Springer Computer Science, Springer, Wien, (2003) 223–228
12. Jedrzejowicz, P., Wierzbowska, I.: JADE-Based A-Team Environment, *Lecture Notes in Computer Science*, Springer Berlin/Heidelberg, 3993 (2006) 719–726
13. Kennedy, J., Eberhart, R. C.: Particle swarm optimisation. *Proc. of IEEE International Conference on Neural Networks*, Piscataway, N.J. (1995) 1942–1948.
14. Laborie, P.: Complete MCS-Based Search: Application to Resource Constrained Project Scheduling, *Proceedings IJCAI-05*, Edinburg, Scotland (2005) 181–186
15. Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*, 2nd Extended Edition. Springer-Verlag, Berlin, Heidelberg, New York (1994)
16. Oster, G.F., Wilson, E.O.: *Caste and Ecology in the Social Insect*. Princeton University Press, Princeton, NJ8 (1978)
17. Parunak, H.V.D.: Agents in Overalls: Experiences and Issues in the Deveopment and Deployment of Industrial Agent-Based Systems. *International Journal of Cooperative Information Systems* **9** (3) (2000) 209–228.
18. PSPLIB, <http://129.187.106.231/psplib>
19. Rabak, C.S., Sichman, J.S.: Using A-Teams to optimize automatic insertion of electronic components, *Advanced Engineering Informatics* **17** (2003) 95–106.
20. Rachlin, J., Goodwin, R., Murthy, S., Akkiraju, R., Wu, F., Kumaran, S., Das, R.: A-Teams: An Agent Architecture for Optimization and Decision-Support. J.P. Muller et. al. (Eds.): *ATAL'98*, Springer-Verlag ,LNAI **1555** (1999) 261–272
21. Reynolds, R.G.: An Introduction to Cultural Algorithms in: A.V. Sebald, L.J. Fogel (eds.). *Proc. 3rd Annual Conference on Evolutionary Programming*, World Scientific, River Edge N.J. (1994) 131–139
22. Sprecher, A., Drexl, A.: Solving multi-mode resource-constrained project scheduling problems by a simple, general and powerful sequencing algorithm. *European Journal of Operational Research* **107** (1998) 431–450
23. Talukdar, S., Baerentzen, L., Govek A., Souza, P.: *Asynchronous Teams: Cooperation Schemes for Autonomous, Computer-Based Agents*. Technical Report EDRC 18-59-96, Carnegie Mellon University, Pittsburgh (1996).
24. Valls, V., Ballestin, F., Quintanilla, S.: A Population-Based Approach to the Resource-Constrained Project Scheduling Problem. *Annals of Operations Research* **131** (2004) 305–324

A Model of Non-elemental Associative Learning in the Mushroom Body Neuropil of the Insect Brain

Jan Wessnitzer, Barbara Webb, and Darren Smith

Institute of Perception, Action and Behaviour,
University of Edinburgh, Edinburgh EH9 3JZ, UK
jwessnit@inf.ed.ac.uk
<http://www.ipab.inf.ed.ac.uk>

Abstract. We developed a computational model of the mushroom body (MB), a prominent region of multimodal integration in the insect brain, and tested the model's performance for non-elemental associative learning in visual pattern avoidance tasks. We employ a realistic spiking neuron model and spike time dependent plasticity, and learning performance is investigated in closed-loop conditions. We show that the distinctive neuroarchitecture (divergence onto MB neurons and convergence from MB neurons, with an otherwise non-specific connectivity) is sufficient for solving non-elemental learning tasks and thus modulating underlying reflexes in context-dependent, heterarchical manner.

1 Introduction

Insects are well adapted to their respective ecological niches, but this does not mean (as is often assumed) that they only perform reflexive, hard-wired behaviours. Insects (and other invertebrates) have been shown to have complex and flexible capabilities. For example, honeybees can solve 'delayed match to sample' and 'delayed non-match to sample' tasks [1] and appear to be able to learn concepts such as symmetry [2]. Studying and understanding these competences (which might be considered minimalist solutions for cognition) in invertebrate brains may ultimately help in better understanding more complex vertebrate brains, and in providing useful design methodologies for intelligent robotics.

A large body of evidence suggests that the mushroom body (MB), a distinct region in the insect brain, plays a cardinal role in adaptive behaviour. One of the central functions linked with the MB is olfactory associative learning and memory. However, the MB in many species receives input from a variety of sensory modalities and is involved in multimodal sensory integration. Its roles have been reported to include context generalisation [3] and place memory [4]. Furthermore, there is evidence that MB neurons react differently to self-generated stimuli and other stimuli [5], suggesting proprioceptive or 'efference copy' input. The MB is thus a potential neural substrate for associations and transfer between sensory modalities, underlying context-specific and non-elemental forms

of learning. These non-elemental forms of learning constitute the main interest of this paper.

Thus far, computational models of MB function have been restricted to classification of sensory inputs in open-loop conditions ([6],[7]). In this paper, we develop a MB model that modulates reflexive sensorimotor loops through non-elemental associative learning [8], that is, forms of learning that go beyond simple associations between two stimuli (classical conditioning) or between a stimulus and a response (instrumental conditioning). In non-elemental learning tasks, the stimuli are ambiguously associated with reward or punishment [8]; each stimulus is followed as often by appetitive (+) as aversive (-) reinforcement so that learning requires the context of the stimulus to be taken into account. In negative patterning, the agent has to learn to approach (appetitive action) the single stimuli A and B but retreat (aversive action) from the compound AB. In biconditional discrimination, the agent has to learn to respond appetitively to the compounds AB and CD but aversively to the compounds AC and BD. In feature neutral discrimination, the agent has to learn to respond appetitively to B and AC but aversively to C and the compound AB. In our simulation experiments, we take ‘reinforcement’ and ‘punishment’ to be sensory cues causing different reflex responses (appetitive or aversive); in successful learning, these responses become associated with the appropriate conditioned stimuli.

Table 1. Stimuli-reward combinations in the non-elemental learning tasks

| Non-elemental learning task | Stimuli-reward combinations |
|--------------------------------|-----------------------------|
| Negative patterning | A+ B+ AB- |
| Biconditional discrimination | AB+ CD+ AC- BD- |
| Feature neutral discrimination | AC+ C- AB- B+ |

We propose a minimalist architecture able to modulate reflex behaviours in closed-loop conditions (where the system’s output influences the system’s inputs) for non-elemental learning tasks. In this paper, we show that the general neuroarchitecture of the MB (fan-out and fan-in) is sufficient for explaining the above forms of non-elemental learning. Section 2 describes the simulation framework and Sect. 3 outlines the general architecture (as suggested by neurobiology). Section 4 describes the neural model in detail. The MB model uses a biologically plausible neuron model and synapses obey a local spike-time dependent plasticity rule. Section 5 presents the simulation results. Section 6 concludes and discusses directions for future work.

2 Experimental Setup for Non-elemental Learning

Our experimental set-up was inspired by conditioning paradigms for visual pattern avoidance in flies, in which the animal in a flight simulator learns an appropriate yaw response to a particular visual pattern which is associated with an unpleasant heat beam. In our simulation, the agent has a limited field of

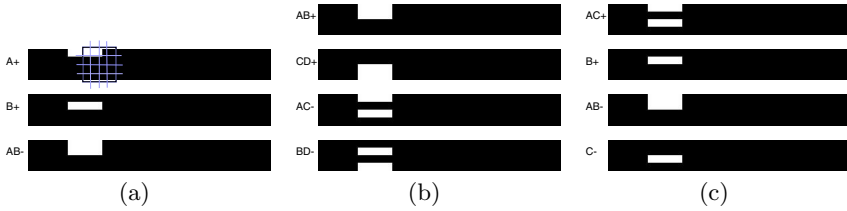


Fig. 1. The wallpapers used for the non-elemental learning tasks: (a) negative patterning, (b) biconditional discrimination, and (c) feature neutral discrimination. The agent’s field of view is represented by the 4-by-4 grid. As the field of view is gradually moved to the left, the visual patterns predict what the agent will experience (+ or -) when it reaches the left edge. Refer to text for further explanations.

view (45 degrees) on a ‘wallpaper’ (of 360 degrees total width) that displays different patterns. In the absence of any action by the agent, the field of view is moved gradually to the left, at 1.5 degrees per millisecond. If it reaches the left edge the agent is “punished” - this generates a reflex action, which moves the field of view back 180 degrees to the right. Before it reaches the edge it will encounter a visual pattern, which can thus be used to predict that the edge will be encountered. The aim is to learn to associate the reflex action with the visual pattern and execute it before encountering the edge, thus avoiding punishment. This anticipatory or conditioned reflex, if executed, will move the field of view 21 degrees to the right.

In the non-elemental learning tasks, there are two reflexes, X0-V0 and X1-V1 (these could be called ‘appetitive’ and ‘aversive’, but in fact they have the same effective result of turning the agent back to 180 degrees). There are two corresponding modes for the simulator, i.e. when the field of view reaches the left edge, the agent experiences either X0 or X1, and will execute the corresponding reflex, V0 or V1. Which experience will occur is predicted by the visual pattern, according to the schemes illustrated in Fig. 1; for example, in negative patterning, the patterns A and B predict X0(+), and the pattern AB predicts X1(-). The agent must learn to execute the correct reflex (V0 or V1) when it sees a particular visual pattern, which will move it away from the edge. If it executes the wrong reflex, then it is instead moved further towards the edge (i.e. 21 degrees to the left).

The field of view contains 45-by-45 pixels, which are mapped onto a 4-by-4 set of sensory neurons (see network description below). White areas on the wallpaper excite these neurons, thus stimulus A will excite the first row of neurons, B the second row and so on. A typical simulation run lasts 50 seconds, during which the wallpapers are switched every 0.5 seconds. The simulation timestep is 0.25ms.

3 The Mushroom Body Model

The neural architecture for the agent is based on the insect brain [9], in particular, on evidence that the MB is involved in modulating more basic, reflexive

behaviours ([10], [11]) and thus acts as a neural substrate for associations underlying context-specific and non-elemental forms of learning. However, the goal of the implemented model design was not to imitate physiological mechanisms involved in MB-mediated learning as closely as possible, but rather to find an abstract description of the underlying principles, able to reproduce associative learning in closed-loop conditions. Yet, we aim to use realistic models of the biological components as more realistic models can be quantitatively and qualitatively different from more abstract connectionist approaches.

Detailed discussion of insect brain architecture is provided in [9]. The main idea is that of parallel pathways, with sensory inputs forming direct reflex loops, but also feeding into secondary routes that are used to place information from various sensory modalities or other domain-specific sensorimotor loops into context. The system can thus improve on reflexive behaviours by learning to adapt and anticipate reflex-causing stimuli. This adaptation process is assumed to occur in the MB, which form such a parallel pathway for sensory inputs in the insect brain (see Fig. 2).

The mushroom bodies in insects have a characteristic neuroarchitecture: namely a tightly-packed, parallel organisation of thousands of neurons, the Kenyon cells. The mushroom bodies are further subdivided into several distinct regions: the calyces (input), the pedunculus, and the lobes (output). The dendrites (inputs) of the Kenyon cells have extensive branches in the calyces, and the axons (outputs) of the Kenyon cells run through the pedunculus before extending to form the lobes. Synaptic interconnections between Kenyon cell axons have been reported [12]. Note that there is considerable divergence (1:50) from a small number of sensory projection neurons (PN) onto the large number of Kenyon cells (KC), and considerable convergence (100:1) from the Kenyon cells onto extrinsic output neurons (EN) (these ratios are estimates based on data from [13]). KC receive direct excitatory input from PN neurons, but also indirect inhibitory inputs from the same neurons via lateral horn interneurons (LHI), arriving shortly after the excitation. These connections are illustrated in Fig. 2.

It is hypothesised that the MB help disentangle spatio-temporal input patterns by operating as coincidence detectors selective to particular correlations in the input spike trains [14]. The mapping of sensory neurons onto MB neurons shows high divergence which can serve for the recognition of unique relationships in primary sensory channels. In our model, nonlinear transformation, separating the activity patterns in the PN layer into sparse activity patterns in the KC layer, is implemented by a randomly determined connectivity matrix between these layers. EN linearly classify the KC activity patterns. Plasticity of KC-EN synapses is achieved with a spike timing dependent plasticity rule. The EN output mediates conditioned responses by activating the appropriate reflex responses. The inhibition from the LHI, quickly following excitation from the PN, limit integration time for the KC to short time windows, making them highly sensitive to precise temporal correlations.

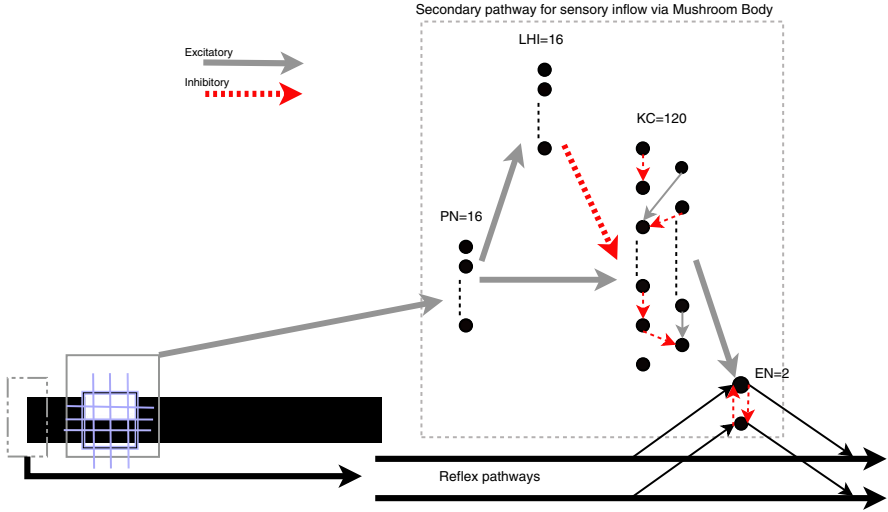


Fig. 2. The implemented MB network receives sensory cues from the visual field via projection neurons (PN), which make direct excitatory connections, and indirect inhibitory connections (via the lateral horn interneurons (LHI)) to the Kenyon cells (KC). The MB output converges on a small number of extrinsic neurons (EN), which are also excited by the underlying direct reflex pathways, and can activate these pathways. Learning occurs between the KC and EN, allowing anticipation of the reflex responses due to associations with particular visual patterns.

3.1 Model Description

We chose the neuron model proposed by Izhikevich [15] since it exhibits biologically plausible dynamics, similar to Hodgkin-Huxley-type neurons, but is computationally less expensive and thus, suitable for large-scale simulation:

$$C \frac{dv}{dt} = k(v - v_r)(v - v_t) - u + I + [\xi \sim N(0, \sigma)] \tag{1}$$

$$\frac{du}{dt} = a(b(v - v_r) - u), \tag{2}$$

where v is the membrane potential and u is the recovery current. $a = 0.3$, $b = -0.2$, $c = -65$, $d = 8$, and $k = 2$ are model parameters. $C = 100$ is the capacitance, $v_r = -60$ is the resting potential, and $v_t = -40$ is the instantaneous threshold potential. ξ is a Gaussian noise term with standard deviation $\sigma = 1$. The variables v and u are reset if $v \geq +35\text{mV}$:

$$\begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases} \tag{3}$$

Synaptic inputs are modelled by:

$$I(t + \Delta t) = gS(t)(v_{\text{rev}} - v(t)), \tag{4}$$

where v_{rev} is the reversal potential of the synapse ($v_{\text{rev}} = 0\text{mV}$ for excitatory and $v_{\text{rev}} = -90\text{mV}$ for inhibitory synapses) and g is the maximal synaptic conductance. $S(t)$ is the amount of neurotransmitter active at the synapse at time t and is updated as follows:

$$S(t + \Delta t) = \begin{cases} S(t)e^{\frac{-\Delta t}{\tau_{\text{syn}}}} + \delta & , \text{ if presynaptic spike} \\ S(t)e^{\frac{-\Delta t}{\tau_{\text{syn}}}} & , \text{ otherwise} \end{cases}, \quad (5)$$

where $\delta = 0.5$ is the amount of neurotransmitter released when a presynaptic spike occurred and τ_{syn} is the synaptic timescale. The simulation timestep Δt is set to 0.25ms.

3.2 Network Geometry

The network geometry as shown in Fig. 2 retains proportional dimensions to the MB system in insects but is smaller in size. A strategy based entirely on random connectivity and self-organisation through local learning and competition is explored. Each neuron pair X-X is connected with probability $p_{X,X}$. The system implements non-specific connectivity with the exception of full inhibitory connectivity between EN (c.f., [8]). We describe the various network layers, their parameters, and their roles below. Learning occurs only through modulation of the KC-EN connections. We report in Sect. 4 the effects on learning performance of changing connectivity between the LHI and KC layers, and varying the size of the KC layer.

PN layer. This layer receives sensory input. The layer consists of 16 neurons (the agent's 45-by-45-pixels FoV is divided into a 4-by-4 grid - c.f., Sect. 2). The input to a single PN neuron is calculated as follows:

$$\text{IPN} = \frac{\frac{\text{sum of pixel values}}{\text{number of pixels}}}{255}. \quad (6)$$

The neurotransmitter released at each timestep is calculated as follows:

$$S(t + \Delta t) = S(t) + \text{IPN} \times \delta. \quad (7)$$

Black areas have a pixel value of 0 whereas white areas have pixel values of 255, thus only white areas in the FoV excite the network.

KC layer. The KC layer consists of ${}^{16}C_2 = 120$ neurons. Each KC will act as a coincidence detector and receive inputs from a small number of PNs ($p_{\text{PN,KC}} = 0.1$). The synaptic timescale $\tau_{\text{PN,KC}}$ is set to 2 ms. This parameter needs to be small in order to make the KC neurons very sensitive to the relative timing of incoming input from the PN layer. This setup allows the KC neurons to act as coincidence detectors. The synaptic strength of PN-KC synapses needed to be carefully adjusted ($g_{\text{PN,KC}}$ are initialised uniformly at random in [20,30]). In addition to this we add an uniformly distributed jitter to the synaptic strengths. We implemented excitatory and inhibitory KC-KC connections ($p_{\text{KC,KC}} = 0.1$,

$\tau_{KC,KC} = 5\text{ms}$) with equal probability ($g_{KC,KC}$ are initialised uniformly at random in $[5,10]$).

LHI layer. Feed-forward inhibition by lateral horn interneurons (LHI) dampens KC activity in the MB. Thus, the integration time for the KC neurons is limited to short time windows, making them highly sensitive to precise temporal correlations. This was implemented through 16 LHIs receiving their inputs from the PN layer and inhibiting activity in the KC layer ($p_{PN,LHI} = 0.2$, $\tau_{PN,LHI} = 5\text{ms}$, $g_{PN,LHI}$ are initialised uniformly at random in $[20,30]$, $p_{LHI,KC} = 0.1$, $\tau_{LHI,KC} = 5\text{ms}$, $g_{LHI,KC}$ are initialised uniformly at random in $[20,30]$).

EN layer. Every KC-EN pair is connected ($p_{KC,EN} = 1$ and $\tau_{KC,EN} = 5\text{ms}$). However, the synaptic conductance $g_{KC,EN}$ for all synapses is initialised to 0, and is subsequently modified by STDP as described below. The ENs also receive excitatory input from the underlying reflex pathways, thus the learning reflects the coincidence of activity in these pathways and particular patterns of KC activity.

3.3 Spike Time-Dependent Plasticity

Synapses are modified using Spike Time-Dependent Plasticity (STDP) which has been observed in biological neural systems (e.g., [16]). In STDP, synaptic change depends on the relative timing of pre- and post-synaptic action potentials. Synaptic conductances are adapted as follows:

$$\Delta g = \begin{cases} A_+ e^{-\frac{t_{\text{pre}} - t_{\text{post}}}{\tau_+}} - \frac{g_{\text{max}}}{r}, & \text{if } t_{\text{pre}} - t_{\text{post}} < 0 \\ A_- e^{-\frac{-(t_{\text{pre}} - t_{\text{post}})}{\tau_-}} & \text{if } t_{\text{pre}} - t_{\text{post}} \geq 0 \end{cases}, \quad (8)$$

where t_{pre} and t_{post} are the spiking times of the pre- and postsynaptic neuron respectively. $A_+ = 2$, $A_- = -1$, $\tau_+ = 50\text{ms}$, and $\tau_- = 5\text{ms}$ are parameters. We modified the STDP rule proposed by [17] by adding an additional term $-\frac{g_{\text{max}}}{r}$ if $t_{\text{pre}} - t_{\text{post}} < 0$ where $r = 10^3$ is a parameter. This means that if postsynaptic spikes are not matched with presynaptic ones, the synaptic conductance between them is decreased by this term. If this modification rule of synaptic conductances g pushes the values out of the allowed range $0 \leq g \leq g_{\text{max}}$, g is set to the appropriate limiting value ($g_{\text{max}} = 30$).

A ‘forgetting’ factor is introduced in the form of a slow decay of g :

$$g(t + \Delta t) = g(t) e^{-\frac{\Delta t}{\tau_{\text{decay}}}} \quad (9)$$

where $\tau_{\text{decay}} = 10^5$.

4 Non-elemental Discrimination Performance

The system was able to learn each of the non-elemental associations shown in Fig. 1. As the system learns to respond to the visual patterns, the reflex responses

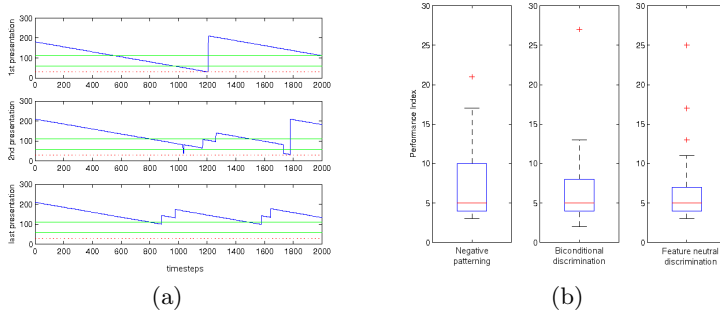


Fig. 3. (a) Agent behaviour in response to first (0ms), second (2000ms) and final (18000ms) presentations of the same wallpaper during a 20s simulation run. At the start it reaches the edge position (lower dotted line) and performs a reflex turn. In the next presentation it responds to the visual stimulus (between the upper dotted lines) but the response is sometimes incorrect. By the final presentation it reliably responds to the visual stimulus and thus successfully avoids the edge. (b) Boxplots of performance for different learning tasks (1) negative patterning, (2) biconditional discrimination, and (3) feature neutral discrimination. The agent anticipates a median of 5 punishments before successfully using the visual patterns to anticipate and avoid it. The simulation runs lasted 50s.

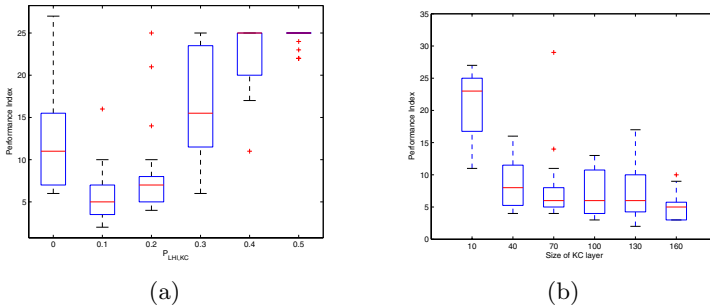


Fig. 4. Boxplots of (a) performance with varying probability of connectivity $p_{LHI,KC}$ between the LHI-KC layers. Performance with varying KC layer size (10,40,70,100,130, and 160 neurons) for a probability of connectivity $p_{LHI,KC} = 0.0$.

to encountering the edge are executed less often and the MB drives the agent’s behaviour (as shown in Fig. 3(a)). The performance index used in this paper is simply the number of times the reflexes are executed. As shown in Fig. 3(a), the naive system will repond with one reflex action during one presentation of one wallpaper. In the biconditional discrimination setup, for example, there are $N_w = 4$ wallpapers which are interchanged every $t_c = 0.5s$ during $t_T = 50s$. Thus, $t_T/(N_w \times t_c) = 25$ activations would mean that a run was unsuccessful. Figure 3(b) shows boxplots of the number of times reflex pathways were active

over 30 simulation runs (each lasting 50 seconds) for each of the three conditioning paradigms. All simulation runs for negative patterning were successful and only one simulation run for biconditional and feature neutral discrimination each was unsuccessful. The agent learnt after a median of 5 activations of the reflex pathways which reflex to use, in response to which visual patterns, to successfully avoid the edge.

Figure 4(a) shows the learning performance with varying probability of connectivity $p_{LHI,KC}$. As the connectivity between LHI-KC neurons increases (and with it the inhibition from this layer), the learning performance becomes maximal at $p_{LHI,KC} = 0.1$. As the inhibition increases further, the performance drops off. With increasing inhibition by the LHI neurons, the activity in the KC layer becomes sparser. Figure 4(b) shows the network performance with varying KC-layer size with probability of connectivity $p_{LHI,KC} = 0.0$. The performance tends to improve with increasing KC layer size.

5 Discussion

Our aim in this paper is to explore the capabilities of an insect-inspired brain architecture, consisting of reactive behaviours which are modulated by the MB. We adapted and simulated a widely used conditioning paradigm (the *Drosophila* flight simulator) for non-elemental tasks of visual pattern avoidance, and tested the role of the neuroarchitecture of the MB for this task. The distinct neuroarchitecture of the MB, where input (PN) neurons diverge onto large numbers of mushroom body (KC) neurons and output converges onto a relatively small number of output (EN) neurons, is able to recognise unique relationships of excitations in the primary sensory channels indicating particular stimulus situations. The output (EN) neurons, receiving the output of mushroom body neurons, mediate conditioned responses.

Based on the proposed architecture and the presented simulation experiments, the following claims can be made. First, the neuroarchitecture is suited for pattern recognition and was successfully demonstrated for non-elemental learning tasks. Note that these tasks essentially require the agent to learn to associate different patterns of stimuli, rather than single stimuli, with the correct action. Hence the pattern recognition properties of the MB could indeed be a suitable substrate for this form of learning (c.f., [8]). Second, coincidence detection and sparse coding are useful for learning. We could show that learning performance is maximal for small levels of inhibition from the LHI layer. Sparse coding of sensory inputs helps “*make neurons more selective to specific patterns of input, hence making it possible for higher areas to learn structure in the data*” [18]. Future work will further investigate these issues for more specific (i.e. non random) connectivity between layers and for modulating larger collections of reflex behaviours.

Acknowledgements. We acknowledge support from EU grant IST-004690 SPARK.

References

1. Giurfa, M., Zhang, S., Jennett, A., Menzel, R., & Srinivasan, M. (2001) The concepts of sameness and difference in an insect. *Nature*. **410**:930-933.
2. Giurfa, M., Eichmann, B., & Menzel, R. (1996) Symmetry perception in an insect. *Nature* **382**:458-461.
3. Liu, L., Wolf, R., Ernst, R., & Heisenberg, M. (1999) Context generalisation in *Drosophila* visual learning requires the Mushroom Bodies. *Nature* **400**:753-756.
4. Mizunami, M., Okada, R., Li, Y., & Strausfeld, N. (1998) Mushroom bodies of the cockroach: their participation in place memory. *J. Comp. Neurol.* **402**:520-537.
5. Mizunami, M., Okada, R., Li, Y., & Strausfeld, N. (1998) Mushroom bodies of the cockroach: activity and identities of neurons recorded in freely moving animals. *J. Comp. Neurol.* **402**:501-519.
6. Nowotny, T., Rabinovich, M., Huerta, R., & Abarbanel, H. (2003) Decoding temporal information through slow lateral excitation in the olfactory system of insects. *J. Comput. Neurosci.* **15**:271-281.
7. Nowotny, T., Huerta, R., Abarbanel, H., & Rabinovich, M. (2005) Self-organization in the olfactory system: one shot odor recognition in insects. *Biol. Cybern.* **93**:436-446.
8. Giurfa, M. (2003) Cognitive neuroethology: dissecting non-elemental learning in a honeybee. *Curr. Opin. Neurobiol.* **13**:726-735.
9. Wessnitzer, J. & Webb, B. (2006) Multimodal sensory integration in insects - towards insect brain control architectures. *Bioinspir. Biomim.* **1**:63-75.
10. Okada, R., Sakura, M., & Mizunami, M. (2003) Distribution of dendrites of descending neurons and its implications for the basic organisation of the cockroach brain. *J. Comp. Neurol.* **458**:158-174.
11. Menzel, R. & Giurfa, M. (2001) Cognitive architecture of a mini-brain: the honeybee. *Trends Cogn. Sci.* **5**:62-71.
12. Heisenberg, M. (1998) What do the mushroom bodies do for the insect brain? An introduction. *Learning Memory* **5**:1-10.
13. Laurent, G. (2002) Olfactory network dynamics and the coding of multidimensional signals. *Nature* **3**:884-895.
14. Perez-Orive, J., Bazhenov, M., & Laurent, G. (2004) Intrinsic and circuit properties favor coincidence detection for decoding oscillatory input. *J. Neurosci.* **24**:6037-6047.
15. Izhikevich, E. (2006) Dynamical systems in neuroscience: the geometry of excitability and bursting. The MIT Press.
16. Dan, Y. & Poo, M. (2004) Spike timing dependent plasticity of neural circuits. *Neuron* **44**:23-30.
17. Song, S., Miller, K., & Abbott, L. (2000) Competitive Hebbian learning through spike-timing dependent synaptic plasticity. *Nature Neurosci.* **3**:919-926.
18. Olshausen, B. & Field, D. (2004) Sparse coding of sensory inputs. *Curr. Opin. Neurobiol.* **14**:481-487.

Performance-Based Bayesian Learning for Resource Collaboration Optimization in Manufacturing Grid*

Jian Zhou^{1,2}, Qing Li¹, Jim Browne³, Qing Wang¹, Paul Folan³, and TianYuan Xiao¹

¹ Department of Automation, Tsinghua University, Beijing, 100084, P.R. China
j-zhou02@mails.tsinghua.edu.cn, zhoujiandragon@gmail.com

² National University of Defense Technology, Changsha, 410073, P.R. China

³ Computer Integrated Manufacturing Research Unit, National University of Ireland, Galway, Ireland

Abstract. Following the rapid development of Grid computing, Grid technology has been introduced into the manufacturing realm and is contemporarily being considered for the sharing of manufacturing resources. However current research in the subject-area is still immature and mainly focuses on conceptual framework development. Here a concrete performance-based Bayesian method for resource collaboration optimization in Extended Enterprise is proposed which improves and promotes research in applying Grid-thinking in inter-organizational manufacturing value chains. Based on the research background, problem statement, and the consideration of Bayesian learning, the method for probability dependency relationship modeling between the performance values of different manufacturing resource nodes in the Extended Enterprise is analysed; and is subsequently complimented by the development of an extended method for more general use. Finally, a system dynamics simulation model for the proposed method is established and the validity and effectivity of the suggested method is tested via a simple case study.

1 Introduction

The contemporary manufacturing environment has seen the rise of a set of problems that require the individual firm to satisfy what Browne et al. [1] suggests are issues of global marketing, organizational structuring, and product and process development. Each of these issues in turn has occasioned a closer examination into how a tighter collaboration of independently-owned firms may be achieved. Such analysis has given rise to the Extended Enterprise paradigm and other collaborative manufacturing rationales. The Extended Enterprise (EE) is a formation of closer co-ordination in the design, development, costing and the co-ordination of the respective manufacturing schedules of co-operating independent manufacturing enterprises and related suppliers [8]; and is the consequent result of a move away from the traditional view of manufacturing companies with clear boundaries, limited relationships with other

* Sponsored by China 863 Program, No. 2001AA415340; Sponsored by China National Natural Science Fund, No. 60474060.

companies and a focus on internal efficiency and effectiveness only [2]. The EE concept places emphasis on resource-sharing enterprise partnerships to facilitate both integration and co-operation across the value chain; and in doing so, signally influences approaches towards greater manufacturing collaboration, especially by avowing a greater usage and sharing of contemporary information and communication technologies.

Advances in Grid computing technologies fall into this latter category and make them ideal for application in EE. The early development of Grid computing technologies was motivated by the problems of creating scientific resource-sharing applications, and increasing the functionality and availability of these by coupling heterogeneous entities such as scientific instruments, remote computers and archives [6][10]. With urgent requirements on manufacturing firms to increase collaborative activities with industrial partners, Grid may be adapted to the needs of manufacturers to help meet these goals. The concept of the “Manufacturing Grid” (MG) [5] is thus proposed with reference to the idea and application of a Computing Grid. This relatively new concept is explored further below where we propose a performance-based Bayesian method for resource collaboration optimization in the EE MG.

2 Research Background and Related Work

In the EE MG, different resources exist between computing and manufacturing, while the key one is manufacturing resources (MR) which are kinds of manufacturing capability and capacity of certain individuals or organizations, such as design, supply and production. The performance value here is concerned with the related performance of the manufacturing individual or organization, and denotes the relevant value of capability and capacity to supply the MR or the demands for them as well as the quality and the time consumed during their use. Further, resource collaboration is supposed to happen between different individuals or organizations and thus there exist certain relationships between them. Thus the individual, unit or organization with certain MR in the collaboration can be denoted as a MR node. These form a MR network in the EE MG, which may similarly be depicted in different forms [9].

Here, the method is presented mainly based on the performance values of different resource nodes. Specifically, the relationships between resource nodes discussed principally are the MR dependency relationship (DR); that is, the relationships between resource nodes in an EE collaboration are dependent as downstream partners rely on the delivery of upstream MR to compete. The DR proposed in this paper is a kind of representation form of a causal relationship in a sense. The causal relationship between nodes and their related performance is a well-researched topic. The models of causal relationship were introduced into the scope of enterprise system engineering by Cooper [4]; while Ronald and Gyula [11] systematically summarized and combined a serial of methods such as causal structure presented in a graphical way, statistical methods, system dynamics and qualitative reasoning to model and analyze the causal relationship between the performances of nodes.

In the study of relationships between node's performance, the introduction of certain fixed weights or fuzzy coefficients usually can be used to improve the qualitative relationship. But in the practical engineering application, it is found that

not only does there exist a correlation between the performance values of nodes with DR, but the measure w of this relationship, like a weight or fuzzy coefficient, is related to the performance value of the node as well. For example, supposing that there are two nodes A and B , and B depends on A ; when the performance value of A has reached a certain level, further performance improvement of A will not lead to a corresponding performance improvement of B . However, when decreasing the performance value of A to a certain level, its influence on the performance of B will become, suddenly, considerable.

The common methods of researching the relationship between performances of nodes, such as system dynamics and fuzzy cognitive mapping, are mainly based on fixed relationships between them, in which the relationship measure w has definite value. So it cannot reflect the above described relationship between measure w and the performance value of node d . To resolve this kind of problem and propose a more convincing and effective method for the MR collaboration, we introduce conditional probability relationships into the DR modeling of node's performances--known as probability cognitive mapping in cognitive research [7].

3 Probability DR Modeling for MR Collaboration

3.1 Basic Resources Nodes and Probability DR Model

Generally speaking, a MR network model RM can be represented as a triple $RM = \langle R, RL, PI \rangle$, where R is the set of resource nodes (RN) and can be described as $R = \{RN_i, i = 1, \dots, N\}$; where N is the number of resource nodes; RL denotes the set of relationships between resource nodes; and PI is the set of performance information value (d) that denotes the performance information of the resource node. Thus the performance status of the resources network at a time t can be implied by the information in PI . PI can be shown as the following function:

$$PI(t) = f(d_i, p), RN_i \in R, i = 1, \dots, N \tag{1}$$

where d_i is the performance information value of some resource node RN_i , and p , which belongs to RL , shows the relation between d_i of different nodes. So p_{ij} is used to denote the relationship between nodes RN_i and RN_j : $p_{ij} = f(d_i, d_j)$. That is:

$$p_{ij} : d_i = f \left(\sum_{\substack{j=1 \\ j \neq i}}^N d_j w_{ji} \right) \tag{2}$$

which reflects the set of resource nodes $\{d_j\}$ having a relationship with d_i at the time t ; and w_{ji} is the measure of this kind of relationship in terms of weights or fuzzy related coefficient.

As discussed before, w_{ji} is usually related with d_j , i.e. when the value of d_j varies at different time, the value of w_{ji} will change with it. This indicates that the value of w_{ji} is not stable. Therefore we can exploit the conditional probability relationship to represent this kind of phenomenon:

$$d_i = f \left[\sum_{\substack{j=1 \\ j \neq i}}^N d_j w_{ji} P(w_{ji} | d_j) \right] \tag{3}$$

where $P(w_{ji} | d_j)$ shows the uncertainty of w_{ji} caused by d_j . Obviously when w_{ji} is not influenced by d_j , Equation (3) can be simplified as Equation (2).

Note that the DR model of resource nodes described by Equation (3) not only includes the usual description of the relationship measure between resource nodes, but also indicates the dynamic aspect of this kind of relationship measure; hence it has an increased ability to describe the relationship between nodes. In particular, the probability DR model in Equation (3) is not about the conditional probability relationship between two resource nodes $P(d_i | d_j)$, but between the measure w_{ji} and the value of performance information $P(w | d)$.

3.2 Extended Model of Probability DR Between Resource Nodes

In the basic model, as shown in Equation (3) above, the DR between measure w_{ji} and the value of performance information d_j is discussed. From this we can improve and extend it to enhance its representational ability for the probability DR between resource nodes.

(1) The measure w_{ji} is influenced by more factors

In addition to d_j , more factors will influence the value of w_{ji} . Therefore more generally, the value of w_{ji} can be affected by the performance information of any other resource node other than RN_i and RN_j . Equation (3) can be extended as:

$$\left\{ \begin{aligned} d_i &= f \left[\sum_{\substack{j=1 \\ j \neq i}}^N d_j w_{ji} P(w_{ji}) \right] \\ P(w_{ji}) &= P(w_{ji} | d_j) + P(w_{ji} | d_m) + \dots P(w_{ji} | d_n) \\ d_m, d_n &\in PI \text{ and } m, n \neq i, j \end{aligned} \right. \tag{4}$$

in which d_m, d_n are other factors of the model RM influencing w_{ji} besides RN_i and RN_j .

Thus the above extended model can indicate that the relationship between resource nodes is influenced by other factors in a more extensive way. Also, in the model of the MR network, there is always lack of methods to quantitatively formalize the relationship between resource nodes; so, based on the model given in Equation (4), a method which includes the probability distribution relationship between resource nodes is proposed to resolve this problem. If the measure of the relationship RL_{ji} and its probability influence by every factor d_n can be obtained by some certain methods (such as the method based on apriori knowledge), the probability distribution of every influencing factor can be calculated as per Bayes Theory:

$$P(d_n) = \frac{P(w_{ji} | d_n)}{\sum_{m=1}^N P(w_{ji} | d_m)} = \frac{P(w_{ji} | d_n)}{\|RL_{ji}\|}, \quad n = 1, \dots, N, \tag{5}$$

where $\|RL_{ji}\|$ denotes the measure of relationship RL_{ji} determined by some kind of methods beforehand; and where the relationship RL_{ji} has exceptional phenomenon, $P(d_n)$ is the abnormality probability of the factor d_n which influences RL_{ji} . In this way, Equation (5) provides a method to reason and analyze abnormal events based on the already built model of probability DR between resource nodes.

(2) Model of probability DR between resource nodes with temporal characteristics

By analogy with the fuzzy cognitive models considering temporal aspects proposed in the literature [7], we can improve the model in Equation (3) and form the following model with temporal memory effect:

$$d_i(t+1) = f \left[\sum_{\substack{j=1 \\ j \neq i}}^N d_j(t) * w_{ji}(t) * P(w_{ji}(t) | d_j(t)) + c * d_i(t) \right] \tag{6}$$

where $d_i(t)$ and $d_j(t)$ denote the values of performance information of resource nodes RN_i and RN_j at time t ; and $d_i(t+1)$ is that of resource node RN_i at time $t+1$. Then $P(w_{ji}(t) | d_j(t))$ represents the uncertainty of $w_{ji}(t)$ influenced by $d_j(t)$. The term $c*d_i(t)$ means the influence of the performance information value at the next time exerted by the value of the performance information in the previous time and c is the influencing factor.

Thus the model in Equation (6) not only illustrates that the measure of the relationship between resource nodes is dynamically affected by the value of performance information, but also implies a variation with time which enhances the representational ability of the dynamic aspect.

Further, supposing in Equation (6) with respect to node RN_i , that d_i is only related with the value of performance information at the previous time—that is to say, conditional probability $P(w_{ji}(t) | d_j(t))$ will not vary with time—then the Equation (6) can be modified as:

$$d_i(t+1) = f \left[\sum_{\substack{j=1 \\ j \neq i}}^N d_j * w_{ji} * P(w_{ji} | d_j) + c * d_i(t) \right] \tag{7}$$

Accordingly all the nodes RN in resources network model influence the relationship RL between the nodes in terms of the Markov Process (if observing the model in discrete time intervals, this process becomes a Markov Chain); this provides a much more general way to study the dynamic aspect of the resources network model.

(3) Adjacency matrix of the model

Exploiting the representational methods in the study of cognitive mapping, we can build the adjacency matrix of the model of probability DR between resource nodes based on Equation (3). Supposing that $d_{i=1,\dots,N}$, $d_i \in PI$ is the value of performance information of $RN_{i=1,\dots,N} \in R$, then:

$$\begin{matrix}
 & d_1 & d_2 & \dots & d_j & \dots \\
 \begin{matrix} d_1 \\ d_2 \\ \vdots \\ d_i \\ \vdots \end{matrix} & \begin{bmatrix} 0 & P(w_{12} | d_1) & \dots & P(w_{1j} | d_1) & \dots \\ P(w_{21} | d_2) & 0 & \dots & P(w_{2j} | d_2) & \dots \\ \vdots & \vdots & \dots & \vdots & \vdots \\ P(w_{i1} | d_i) & P(w_{i2} | d_i) & \dots & P(w_{ij} | d_i) & \dots \\ \vdots & \vdots & \dots & \vdots & \vdots \end{bmatrix}
 \end{matrix} \tag{8}$$

in which if $P(w_{ji} | d_j) = 0$, there is no direct DR between nodes RN_i and RN_j . With reference to methods in fuzzy theory, if the conditional probability relationship $P(w_{ji} | d_j)$ is extended from interval $[0, 1]$ to $[-1, 1]$, then $-P(w_{ji} | d_j)$ indicates that RN_i is relied on by RN_j . Therefore the construction of the adjacency matrix enriches the representational form of the conditional probability relationship and thus the model proposed in this paper can be applied to analyze the generalized relationship in terms of the network or tree structure between resource nodes.

3.3 Application of Performance Modeling Based on Probability DR

In practical application, the above models shown in Equation (3)-(7) mainly require three problems to be resolved: 1. the function f which denotes the relationship between two resource nodes; 2. the measure w of the relationship between two nodes; and 3. how to use the conditional probability $P(w_{ji} | d_j)$.

Firstly the frequently used functions of the relationship representation f are two value functions or probability distribution functions. Regarding the measure of the relationship w , the representation methods, such as certain weight coefficients or fuzzy related functions, are usually applied. Thus the critical existing problem is the last: how to apply the conditional probability relationship $P(w | d)$ between the measure w and the value of node performance information d . Nevertheless we must note that $P(w | d)$ here is in its broad sense, which indicates the relationship measure w is influenced by the value of node performance information d ; this can imply not only the subjective belief degree for this relationship by certain specialists, but also some kind of stable objective probability. The former one is called Apriori Application while the latter is named as Aposteriori Application.

In the aposteriori application, $P(w(t) | d_j(t), \dots)$ is always determined after the resources network model has been put into practice for some time and thus has accumulated data to some extent. Then statistical methods or Bayes equation can be used to obtain this probability relationship. Self-adaptive learning algorithms like cognitive mapping with the self-learning function of the neural network also can be used to solve the probability distribution of the conditional probability relationship.

Opposed to aposteriori application, the application based on apriori knowledge is more critical since it should be completed before the resources network model is put

into practice, which will directly influence the effect of application and probably result in the expense of adjustment afterwards. During apriori application, the conditional probability $P(w(t) | d_j(t), \dots)$ can always be set beforehand according to the engineering experience of related specialists or some heuristic information.

4 Case Study

In this section, a simple case study of the EE will be put forward to illustrate the presented probability DR model for MR collaboration. The illustration is mainly concerned with the basic probability DR model as described in Equation (3), and the illustrated model will be simplified also to make the case study more understandable. Based on industrial practice, a simplified EE process model is presented in Fig. 1, which comprises three MR nodes: the original equipment manufacturer (OEM), the first tier supplier and the second tier supplier; there are three production processes--from the purchase of raw materials to the production delivery for customers--that are actually the production of accessories/components, semi-manufactured goods and the final product.

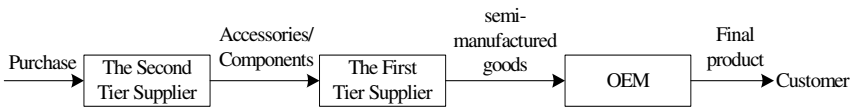


Fig. 1. A simple Extended Enterprise process

Figure 1 is a typical pull production system and, as a result, bullwhip effect exists in it. In this case, it is necessary to optimize the MR collaboration in this EE process, and weaken the fluctuation caused by the abnormal requirements of the node in the downstream position. A related simulation diagram, drawn using the VENSIM application, is shown as Fig. 2. VENSIM is a software for modeling and simulating dynamic systems. It is used for developing, analyzing, and packaging high quality dynamic feedback models. Models are constructed graphically or in a text editor, including features such as dynamic functions, subscripting (arrays), Monte Carlo sensitivity analysis, optimization, data handling, and application interfaces.

The relevant MR nodes and performance factors concerned in the simulation are depicted in Table 1.

The DR model as shown in Fig. 2 has applied the traditional DR modeling based on system dynamics [12]. The adjustable performance factors $EV0$, $EV1$, $EV2$ and $EV3$ transfer the feedback information from downstream to upstream in the EE level by level, which can play an important role in adjusting the quantity of products and lowering its fluctuation. $W0$, $W1$, $W2$ and $EV3$ are the fixed weight coefficients in the transfer process which will not be influenced by the performance value of the transfer relationship. Through the simulation of VENSIM [13], we can show, in the left-hand column in Table 2, the influence from the abnormal change of customer requirements to the quantity of products $R1$, $R2$ and $R3$.

Grounded on the model with probability DR described in Equation (3) before, the above presented model can be improved to reflect the DR between weight coefficient w and the performance value of node. Furthermore, for practical purposes, the application of the model is based on apriori knowledge, while the conditional probability distribution of $W0$ and $EV1$ can be obtained through statistical methods according to different performance values of $EV1$. Accordingly the value of $W0$ can be determined through fuzzy rules. The similar improvement method is also applied to $W1$ and $W2$. We use the improved model to optimize the EE collaboration and through the simulation of VENSIM, we can show, as in the right-hand column of

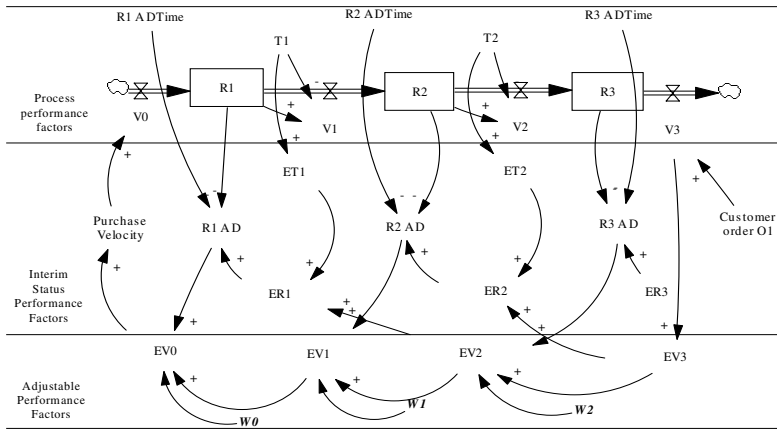


Fig. 2. The simulation for dependency relationship model

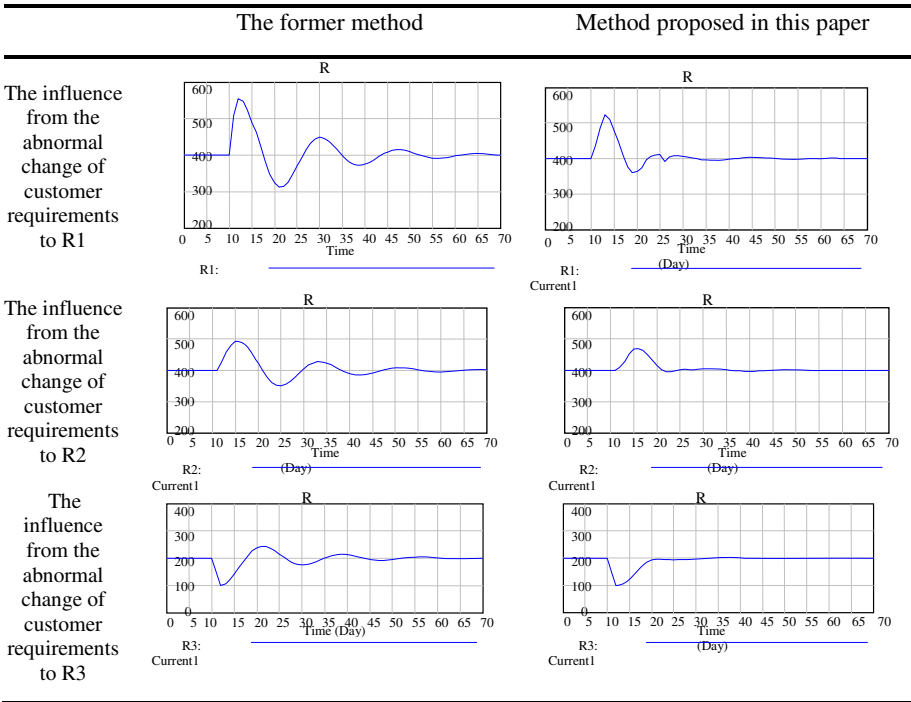
Table 1. Manufacturing resources of nodes and performance factors concerned

| Manufacturing Resources of Nodes | Process Performance Factors | Interim Status Performance Factors | Adjustable Performance Factors |
|--|---|------------------------------------|--|
| R1: The quantity of products of the second tier supplier | V0: Raw materials supply velocity | ET1: Expectation of T1 | EV0: Expectation of the velocity of V0 |
| R2: The quantity of products of the first tier supplier | V1: Accessories /Components supply velocity | ET2: Expectation of T2 | EV1: Expectation of the velocity of V1 |
| R3: The quantity of products of the OEM | V2: Semi-manufactured goods supply velocity | R1 AD: Adjustment of R1 | EV2: Expectation of the velocity of V2 |
| | V3: Final product supply velocity | R2 AD: Adjustment of R2 | EV3: Expectation of the velocity of V3 |
| | T1: Run time of Accessories /Components | R3 AD: Adjustment of R3 | |
| | T2: Run time of Semi-manufactured goods | ER1: Expectation of R1 | |
| | R1 AD Time: Adjustment time of R1 | ER2: Expectation of R2 | |
| | R2 AD Time: Adjustment time of R2 | ER3: Expectation of R3 | |
| | R3 AD Time: Adjustment time of R3 | | |

Table 2, the influence from the abnormal change of customer requirements to the quantity of products $R1$, $R2$ and $R3$.

Thus, as illustrated in Table 2, it can be concluded that, in comparison with common DR model, the improved model can effectively control and monitor the production process, alleviate the fluctuation of the quantity of products, and restrain the bullwhip effect along the value chain to a great extent. In this way, it can improve the effectiveness and efficiency of resource collaboration and decrease the waste of MR.

Table 2. The comparison of effectivity between the former method and the proposed one



In the above case study, the proposed model in this paper acts as a “performance observation instrument” which can supervise the production process. It is actually similar to controlling a system with non-linear factors with probability and statistical methods in control theory. Because it is hard to describe the non-linear aspects of a resource collaboration system, the mathematical model of the system cannot be presented directly and thus be controlled effectively. However the improved model based on conditional probability can statistically observe and monitor this kind of “non-linear factors” in the resource collaboration system very well.

The presented case just discusses a simplified EE model; in reality, the EE will probably display a more complex network structure. However the extended model proposed in this paper can be applied to deal with this kind of complicated situation. In fact, the methods introduced in this paper will have a much more obvious effect and advantages in this context.

5 Conclusions

Following the needs of manufacturers to collaborate more intensively in the EE environment, this paper proposes a MG viewpoint to optimize resource collaboration among partnering firms. Taking into account not only the correlation between MR nodes but also the probability relationship between the measure W and the performance of related nodes, a Bayesian method for the optimization of MR collaboration is outlined to meet these urgent requirements. The proposed method can manage and monitor the DR between different resource nodes more accurately and provide a more convincing reasoning mechanism to find the right cause for implementing adjustments of performance improvements. Accordingly, it can enhance the scalability, flexibility and reliability of the MR configuration and allocation in the collaboration network, and can improve the validity and effectivity of MR collaboration--here tested with a simple simulation case study. Moreover, the method presented will make the management of complex resources collaboration network more computer-processible and reliable so as to make the MG in the EE more feasible.

References

1. Browne, J., Sackett, P. J., Wortmann, J. C.: Future manufacturing systems - towards the extended enterprise. *Computers in Industry*, 25 (1995) 235-254.
2. Browne, J. Zhang, J.: Extended and virtual enterprises - similarities and differences. *International Journal of Agile Management Systems*, 1 (1) (1999) 30-36.
3. Chen Y.: Justification Criteria of CIM Implementation [J]. *Computer Integrated Manufacturing Systems-CIMS*, 3(3), (1997)
4. Cooper, G.: An overview of the representation and discovery of causal relationships using Bayesian Networks-Computation, Causation and Discovery [M]. MIT Press, Menlo Park, CA, (1999) 3-62
5. Fan, Y.S., Zhao, D.Z., Zhang, L.Q., Huang, S.X., Liu, B.: Manufacturing grid: Needs, concept, and architecture. *Grid and Cooperative Computing*, PT 1 3032: (2004) 653-656
6. Foster, I., Kesselman, C., Tuecke, S.: The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of Supercomputer Applications*, 15(3) (2001) 200-222
7. Gao, J.: Theories of intelligent information processing [M]. China Machine Press, Beijing (2004)
8. Jagdev, H. S., Browne, J.: The extended enterprise - a context for manufacturing. *Production Planning and Control*, 9 (3) (1998) 216-229.
9. Jagdev, H.S., Thoben, K.-D.: Typological issues in Enterprise Networks. *Journal of Production Planning and Control*, Vol 12, No 5 (2001)
10. Jun-Jang Jeng, Henry Chang, Jen-Yao Chung.: BPMM: A Grid-Based Architectural Framework for Business Process Meta Management. *Proceedings of the 2003 Symposium on Applications and the Internet (SAINT'03)* (2003)
11. Ronald, D., Gyula, V.: Causal modeling alternatives in operations research: Overview and application [J]. *European Journal of Operational Research*, (2004) 92-109
12. Serman J.: *Business dynamics: systems thinking and modeling for a complex world* [M]. Irwin/McGraw-Hill, Boston, (2000)
13. Ventana Systems Inc. Vensim system [EB/OL], www.vensim.com, (2005)

A Hybrid Simulated-Annealing Algorithm for Two-Dimensional Strip Packing Problem

Türkey Dereli and Gülesin Sena Daş

Department of Industrial Engineering, University of Gaziantep, 27310, Turkey
{dereli, semre}@gantep.edu.tr

Abstract. This paper presents a hybrid simulated-annealing (SA) algorithm for two-dimensional strip packing problem (2D SPP) where a set of small rectangular items has to be allocated on a larger stock rectangle in order to find a minimum height. A new recursive placement procedure is proposed and the procedure is combined with the SA algorithm. The hybrid-SA algorithm was tested on a set of benchmark problems taken from the literature. The computational results have validated the quality of the solutions and usefulness of the proposed hybrid-SA algorithm.

1 Introduction

Cutting and Packing Problems (CPPs) are a set of widely studied problems which are defined as *geometric - combinatorial* problems. CPPs are *geometric-based* because within each large object, one or more small items are arranged in such a way to avoid overlapping and to fit into the object's geometric boundaries. They are also *combinatoric-based* since small items are to be assigned to the large objects. In other words, each large object is assigned a given set of small items and each item is assigned to at most one large object [1].

Two-dimensional rectangular packing problems (2D RPP) which encountered in many industries are a sub-problem of CPPs. Due to the different applications in different industries such as the textile, glass, wood and paper industries, some additional constraints and/or assumptions for the solution of the problem may be needed. Main concentration is generally given to cutting of rectangular shaped items from a large rectangular sheet of material in these industries.

2D RPP can be defined as *allocating a set of items to a single object to minimize the used object space*. The usual objective of the allocation process is to maximize the material utilization and hence, to minimize the "wasted" area [2]. 2D RPP have been studied under two titles; *2D bin-packing problems and 2D strip packing problems*. In case of 2D bin-packing problems (2D BPP); the units are finite rectangles, and the objective is to pack all the items into the minimum number of units, while in case of 2D strip packing problems (2D SPP), there is a single standardized unit of given width, and the objective is to pack all the items to find the minimum height [3]. For more information on packing problems, the readers are referred to [3]-[6].

In this study, 2D SPP with rectangular pieces is considered in order to minimize the "height". The following assumptions are made when solving the problem;

- i) the object has a fixed width and infinite height,
- ii) each item has a fixed width and height,
- iii) moreover, the set (of items) may contain identical items, and
- iv) the items to be packed can be *rotated* by 90° .

A feasible solution of the problem must ensure that there is not an overlap between items.

It should be noted that 2D PP are strongly NP-hard [3]. Although, some exact/optimal algorithms were proposed for the solution of the problem, they might not be practical for large-scale problems in which large number of rectangular items has to be placed [7]. Due to the complex nature of the problem, many heuristic and meta-heuristic packing algorithms have been suggested in the literature for the solution of the 2D PP as well as “*hybrid algorithms*” which hybridize a heuristic placement-policy with meta-heuristics are employed. The ability of heuristics, meta-heuristics and hybrid algorithms to reach near-optimal/optimal solutions in a *reasonable amount of time* encouraged many researchers to apply them to solve large sized/complex packing problems. However when compared to the great amount of work on heuristics for the packing problem, the work on meta-heuristics is limited. The “*computational time*” required to obtain the solution as well as the “*solution quality*” determines the effectiveness of these algorithms. Recently, Zhang et al. [7] have proposed a fast/new recursive algorithm especially for solving large-sized test cases in strip rectangular packing problem. One of the first attempts on using meta-heuristics for the 2D PP is done by Smith [8], who used Genetic Algorithms (GA) with two heuristic packing routines; *slide algorithm and skyline algorithm*. Another popular meta-heuristic algorithm; *Simulated Annealing (SA)*, was used by Downsland [9], for the solution of problems with identical and non-identical items. In 1996, Jacobs [10] hybridized Bottom-Left (BL) heuristic with GA. Another SA algorithm was presented by Lai and Chan [11] for the cutting stock problems. Leung et al. [12] tested a set of combinations of SA and GA with Bottom Left (BL) and Difference Process (DP) heuristics. Their experimental results showed that GA and DP heuristics provide more favorable results for their test data. Hopper and Turton [2] presented an empirical investigation of meta-heuristic and heuristic algorithms for 2D PP. They hybridized two heuristic procedures (BL and Bottom Left Fill algorithm) with three meta-heuristic algorithms (GA, SA, Naïve Evolution) and local search. Unlike Leung et al. [12], SA has achieved the best layout quality.

In this paper, a hybrid simulated-annealing (SA) algorithm for 2D SPP is presented. A new recursive placement procedure is proposed and combined with the SA algorithm. A computational study is also performed in order to validate the quality of the solutions and usefulness of the proposed *hybrid-SA* algorithm.

2 Recursive Placement Procedure for 2D SPP

In this paper, a new recursive placement procedure for 2D SPP is introduced. The main advantage of using *recursive algorithms* is that they reduce the solution to a problem with a particular set of input to the solution of the same problem with smaller input values [13]. Besides, the recursive algorithms are simple and easy to implement. The recursive placement procedure that is employed in this study is as follows;

- Pack the first item into the bottom left corner ((0, 0) coordinate) of the object (This operation also divides the packing space S into two subsequent subspaces).
- Pack the next item into the subspace S_1 . If packing to the subspace S_1 is not possible, then pack the item to the subspace S_2 . Call this procedure recursively until all the items are packed.

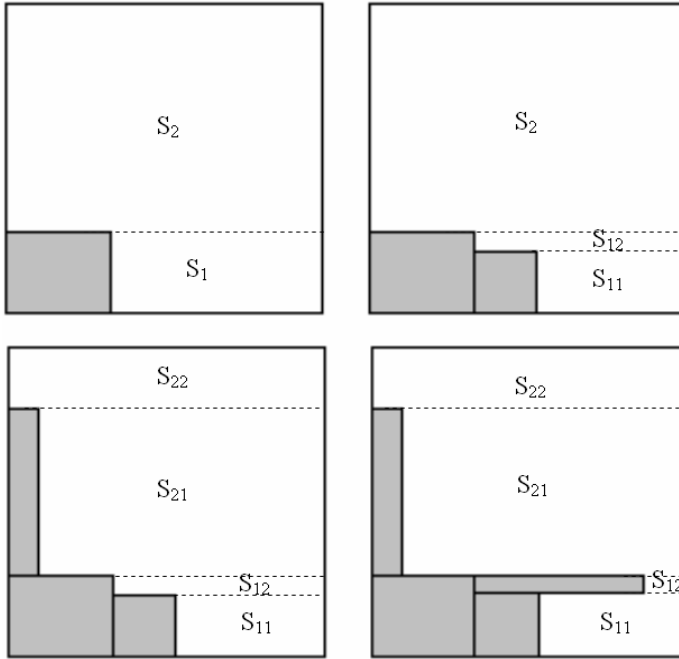


Fig. 1. Packing policy using the recursive placement procedure

In Fig. 1, the working principle of the recursive placement procedure is illustrated. The first item is packed in the bottom-left corner of the larger object. As a result of this packing, two subspaces (S_1 and S_2) are generated. The algorithm tries to pack the next item into the bottom-left corner of the subspace S_1 . If this is not possible, the item is packed to the bottom-left corner of the subspace S_2 . This packing will again divide the subspace in which packing is done into two subspaces. The algorithm will try to pack a new item firstly to subspace S_{11} , then to subspaces S_{12} , S_{21} and to S_{22} . The procedure will be called *recursively* until all the items are allocated into the large object.

In order to demonstrate the difference of the proposed recursive placement procedure from the well known Bottom - Left (BL) heuristic, an example problem from Hopper and Turton [2] is solved using both of the methods. Figure 2 shows how the rectangular pieces described by the permutation (2, 6, 4, 3, 0, 1, 5) are allocated on the rectangular large object.

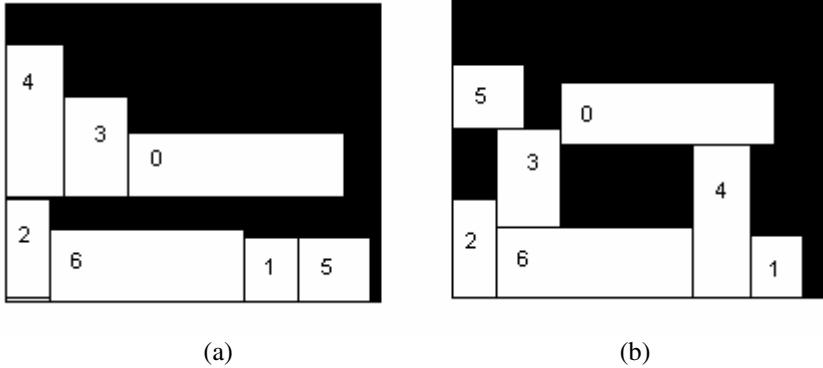


Fig. 2. (a) Packing using the proposed recursive placement procedure (b) Packing using the BL heuristic

3 Simulated Annealing (SA) and Its Hybridization

Simulated Annealing (SA) is a local search meta-heuristic that is applied to many combinatorial optimization problems such as traveling salesman problem, scheduling, graph partitioning, etc. It is introduced by Kirkpatrick et al. [14] as an analogy to the statistical mechanics of annealing in solids.

SA differs from iterative algorithms in that it does not trap into a local optimum but rather reach to global optimum. This is because SA not only accepts neighborhood solutions better than the current solution, but also accepts neighborhood solutions worse than current solution with a probability. This probability which is known as *acceptance probability* is related to the *temperature*, which decreases during the process. As the temperature decreases, the acceptance probability decreases, too. This means that as the temperature decreases, the probability of accepting worse neighborhood solutions decreases. During the annealing process, the temperature decreases gradually. At each temperature, a predetermined number of iterations to search the solution space are conducted. The search terminates when the stopping criteria are met.

In this work, the proposed recursive placement procedure (presented in Sect. 2) is combined with the SA algorithm in order to get *hybrid-SA* algorithm.

2D SPP is modeled as a permutation problem and the SA algorithm is used to search better solutions among the neighbors of the current solution. The neighbor solutions in the algorithm are produced by the mutation operator. This operator is used for either enabling the rotation of an item or disabling of the rotation of an item. After the application of the mutation operator, the items are sorted in decreasing order of height. Then, the sorted list of items is sent to the recursive placement procedure and the “height value” as a result of the allocation of these sorted items is obtained.

The cooling schedule for the parameters of the SA algorithm is adopted from the work presented by Hopper and Turton [2] in order to make a better comparison in between the previous algorithms and the one developed in this study. A proportional decrement is applied to the temperature where rate of cooling (α) is set to 0.987.

Different from the cooling schedule used by Hopper and Turton [2], the starting temperature (T_{in}) is set to 50 (to obtain an initial accepting probability of 80%) and final temperature (T_f) is set to 0.5. The number of iterations (il_{max}) at each temperature is determined as $50N$ total moves or $5N$ successful moves, where N represents the number of items to be placed on the larger object.

A pseudo-code is given below for the *hybrid-SA* algorithm in which the recursive placement procedure (proposed in this study) is embedded into the SA algorithm:

```

Step 1. Generate an initial solution
        Calculate the fitness value  $fitness_0$  using the
        recursive placement procedure;
         $Solution = fitness_0$ ;
Step 2. Parameter initialization;
        2.1. Set the annealing parameters;  $T_{in}, T_f, il_{max}$  and  $\alpha$ .
        2.2. Read the number of items  $N$ .
Step 3. Annealing Schedule;
        3.1. Inner loop initialization;  $il=0$ ;
        3.2. At every temperature achieve equilibrium. Execute inner loop until the condition in 3.2.5 is met;
                3.2.1.  $il = il+1$ ;
                3.2.2. Generate a neighborhood solution and calculate the fitness value  $fitness_{il}$  using the recursive filling procedure;
                3.2.3.  $\epsilon = fitness_{il} - fitness_{il-1}$ ;
                3.2.4. IF ( $\epsilon \leq 0$ ) OR ( $Random(0,1) \leq e^{(-\epsilon/T_{iter})}$ ) THEN accept the new solution,  $Solution = fitness_{il}$ ; ELSE reject the new solution,  $Solution = fitness_{il-1}$ ;
                3.2.5. IF ( $il \geq il_{max}$ ) or  $5N$  successful moves are achieved THEN terminate inner loop and GOTO step 3.3 ELSE continue inner loop and GOTO step 3.2.1
        3.3.  $T_{iter+1} = \alpha * T_{iter}$ ;
        3.4. IF ( $T_{iter+1} < T_f$ ) THEN terminate inner loop and GOTO step 4 ELSE continue inner loop and GOTO step 3.1
Step 4. Terminate the best solution obtained and stop.

```

4 Computational Results

The proposed *hybrid-SA* algorithm is tested on 7 test problems given in [2]. The optimal solutions of these problems (shown in Table 1) are known and each of the test problems has 3 instances which were presented in [2]. The *hybrid-SA* algorithm was

Table 1. Test problems

| Problem category | Number of items | Optimal height | Object dimensions |
|------------------|-----------------|----------------|-------------------|
| C1 | 16 or 17 | 20 | 20 × 20 |
| C2 | 25 | 15 | 40 × 15 |
| C3 | 28 or 29 | 30 | 60 × 30 |
| C4 | 49 | 60 | 60 × 60 |
| C5 | 72 or 73 | 90 | 60 × 90 |
| C6 | 97 | 120 | 80 × 120 |
| C7 | 196 or 197 | 240 | 160 × 240 |

implemented in C++ language and the problems are run on a computer with 797 MHz, Intel Pentium III.

For checking the performance of the proposed *hybrid-SA* algorithm, both average (of 10 simulations) and best values were recorded in this study. The computational results are presented in Table 2 and 3. In Table 2, the relative distance of the solution to the optimum height for each problem is presented. To demonstrate the performance of the *hybrid-SA* algorithm, the computational results on the same test problems found by different researchers are also presented in the same table (Table 2). The

Table 2. Computational results for the test problems (% over optimum)

| | C1 | C2 | C3 | C4 | C5 | C6 | C7 | Avg. |
|---|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| <i>GA + BL</i> (average of 10 runs) | 6 | 10 | 8 | 9 | 11 | 15 | 21 | 11,43 |
| <i>GA + BLF</i> (average of 10 runs) | 4 | 7 | 5 | 3 | 4 | 4 | 5 | 4,57 |
| <i>SA + BL</i> (average of 10 runs) | 4 | 7 | 7 | 6 | 6 | 7 | 13 | 7,14 |
| <i>SA + BLF</i> (average of 10 runs) | 4 | 6 | 5 | 3 | 3 | 3 | 4 | 4 |
| <i>HR</i> (single run) | 8,33 | 4,45 | 6,67 | 2,22 | 1,85 | 2,5 | 1,8 | 3,97 |
| *Hybrid-SA (average of 10 runs) | 1,66 | 4,88 | 4,00 | 3,94 | 3,18 | 3,30 | 3,38 | 3,48 |
| *Hybrid-SA (best of 10 runs) | 1,66 | 4,44 | 3,33 | 3,33 | 2,59 | 2,77 | 3,20 | 3,05 |

Table 3. Running time for the test problems (in seconds)

| | C1 | C2 | C3 | C4 | C5 | C6 | C7 | Average |
|-------------------|-------------|-------------|--------------|--------------|---------------|--------------|---------------|---------------|
| <i>GA + BL</i> | 2,3 | 3,69 | 4,15 | 11,07 | 18,46 | 30,92 | 106,15 | 25,25 |
| <i>GA + BLF</i> | 4,61 | 9,22 | 13,83 | 59,93 | 165,96 | 396,46 | 3581,97 | 604,57 |
| <i>SA + BL</i> | 1,84 | 6,46 | 8,3 | 34,61 | 78,46 | 143,07 | 540 | 126,77 |
| <i>SA + BLF</i> | 3,227 | 11,064 | 18,44 | 152,13 | 530,15 | 1761,02 | 19274,41 | 3107,2 |
| <i>HR</i> | 0 | 0 | 0,03 | 0,14 | 0,69 | 2,21 | 36,07 | 5,59 |
| *Hybrid-SA | 3,64 | 6,36 | 18,78 | 101,2 | 287,27 | 757,2 | 1650,6 | 403,57 |

results are compared with the *GA with BL heuristic (GA+BL)*, *GA with BLF heuristic (GA+BLF)*, *SA with BL heuristic (SA+BL)*, *SA with BLF heuristic (SA+BLF)* reported in [2] and *Heuristic Recursive (HR)* reported in [7]. Running times (in seconds) of all the compared algorithms are also given in Table 3.

For the test problems reported in [2], the *hybrid-SA* algorithm approached to the optimum values with deviations ranging from 1.66% to 4.88% with an average of 3.48%. For the test instances C1(P1), C1(P3) and C2(P3), the *hybrid-SA* algorithm finds “optimal solutions” (notice that the values in parentheses denote “instances”). The packing patterns for these instances C1(P1), C1(P3), C2(P3) and C2(P2) are presented in Fig. 3 and Fig. 4, respectively. The line *in the right of the packing patterns* shown in Fig. 3 and Fig. 4 shows the “optimum height”.

When compared to the *GA+BL* and *GA+BLF*, the proposed *hybrid-SA* algorithm (except for problem C4 solved with *GA+BLF*) produces better results. The improvement on the solution quality, as compared to *GA+BL* and *GA+BLF*, is 70% and 24%, respectively.

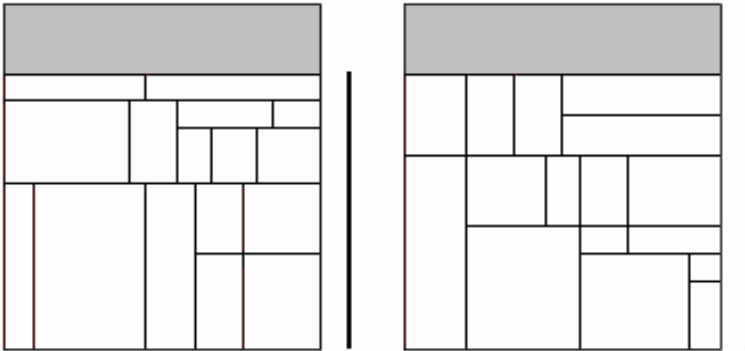


Fig. 3. Optimum packing patterns for C1(P1) and C1(P3) found by *hybrid-SA*

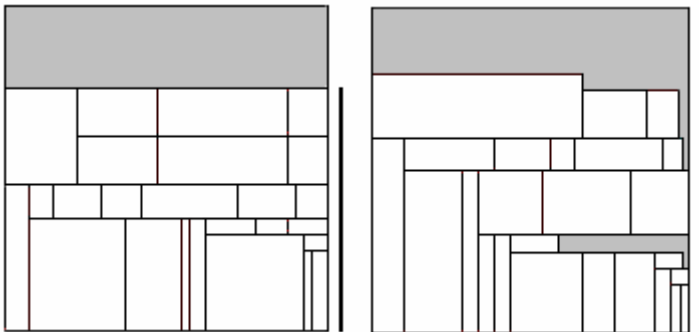


Fig. 4. Optimum packing pattern for C2(P3) and the best solution for C2(P2) by *hybrid-SA*

When compared to the *SA+BL*, the *hybrid-SA* packs better than *SA+BL* for all test cases. The best results obtained by the *hybrid-SA* are better than the *SA+BLF* (except for problem C4). The average results obtained by the *hybrid-SA* also outperform the *SA+BLF* for the problems C1, C2, C3 and C7. The improvement on the solution quality, as compared to the *SA+BL* and *SA+BLF*, is 51% and 13%, respectively.

When compared to the *HR* algorithm, the *hybrid-SA* packs better for the problems C1, C2 and C3. As the number of rectangular items to be packed increases, the *HR* produces better results. However, the *average of deviations from the optimum height* of the proposed *hybrid-SA* algorithm is 12% better than that of *HR* algorithm.

5 Conclusion

A *hybrid-SA* algorithm for 2D packing problems is presented in this paper. A new recursive placement procedure is proposed and integrated with the SA algorithm. The proposed *hybrid-SA* algorithm has been tested on a number of test cases taken from the literature and the results are compared with those of other approaches (*GA+BL*, *GA+BLF*, *SA+BL*, *SA+BLF*, *Heuristic Recursive*).

The computational results show that the proposed SA algorithm outperforms GA (*GA+BL*, *GA+BLF*). When compared to the previously proposed SA algorithms, the best solution and the average solution found by the *hybrid-SA* always outperforms *SA+BL*. The best results obtained by the *hybrid-SA* are better than the **SA+BLF** (except for problem C4) whereas the average results obtained by the *hybrid-SA* outperform the *SA+BLF* for problems C1, C2, C3 and C7.

The findings show that although simple in nature, the proposed recursive placement procedure is quite effective to improve the solution quality of packing problems. When combined with a meta-heuristic such as SA, it is capable of producing good-quality packing patterns. Another advantage of the proposed *hybrid-SA* algorithm is that, it provides promising results within a reasonable amount of computational time. The developed *hybrid-SA* has been also used in a furniture company with positive results. The future work can be done in order to improve the performance of the algorithm using different operators to determine neighbor solutions especially for large-scale problems.

References

1. Dyckhoff, H, Finke, U: Cutting and packing in production and distribution. Physica-Verlag, (1992).
2. Hopper, E, Turton, B,C,H: An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. European Journal of Operational Research, 128, 34-57, (2001).
3. Lodi, A, Martello, S, Monaci, M: Two-dimensional packing problems: a survey. European Journal of Operational Research, 141, 241-252, (2002).
4. Dyckhoff, H: A typology of cutting and packing problems. European Journal of Operational Research, 44, 145-159, (1990).
5. Downsland, K,A, Downsland, W,B: Packing Problems. European Journal of Operational Research, 56, 2-14, (1992).

6. Lodi, A, Martello, S, Vigo, D: Recent advances on two-dimensional bin packing problems. *Discrete Applied Mathematics*, 123, 379-396, (2002).
7. Zhang, D, Kang, Y, Deng, A: A new heuristic recursive algorithm for the strip rectangular packing problem. *Computers & Operations Research*, 33, 2209-2217, (2006).
8. Smith, D: Bin-packing with adaptive search. In: Grefensstette J. (ed.), *Proceedings of an International Conference on Genetic Algorithms and the Applications*. Lawrence Erlbaum: London; 1985, 202-206.
9. Downsland, K,A: Some experiments with simulated annealing techniques for packing problems. *European Journal of Operational Research*, 68, 389-399, (1993).
10. Jakobs, S: On genetic algorithms for the packing of polygons. *European Journal of Operational Research*, 88, 165-181, (1996).
11. Lai, K,K, Chan, J,W,M: Developing a simulated annealing algorithm for the cutting stock problem. *Computers & Industrial Engineering*, 32, 115-127, 1997.
12. Leung, T,W, Yung, C,H, Troutt, M,D: Applications of genetic search and simulated annealing to the two-dimensional non-guillotine cutting stock problem. *Computers & Industrial Engineering*, 40, 201-214, (2001).
13. Rosen, K,H: *Discrete mathematics and its applications*. 5th ed.. McGraw Hill, (2003).
14. Kirkpatrick, S, Gelatt, C,D, Vecchi, M,P: Optimization by simulated annealing. *Science*, 220, 671-680, (1983).

Handling Linguistic Values in Knowledge Acquisition

Dae-Young Choi

Dept. of MIS, Yuhan College,
Koean-Dong, Sosa-Ku, Puchon, Kyoungki-Do, 422-749, Korea
dychoi@yuhan.ac.kr

Abstract. An algorithmic approach for handling linguistic values defined in the same linguistic variable is proposed. It can explicitly capture the differences of individuals' subjectivity regarding linguistic values. The proposed approach can be employed as a useful tool for discovering hidden relationship among linguistic values. Thus, it provides a basis for improving the precision of knowledge acquisition in handling linguistic values. We apply the proposed approach to a collective linguistic assessment among multiple experts.

1 Introduction

Linguistic assessment for selection and ranking is closely related to common facets of human decision making activities in practice. In an academic institution, business organization, awarding/rewarding/funding agencies, etc., most of the decision activities involve a process of linguistic assessments in evaluation. In reality, neither all of the queries nor the corresponding expert opinions are clear and unambiguous. In most of the cases, expert opinions rather come in linguistic forms containing a lot of subjectivity, vagueness and ambiguity too. The main problem in such cases is the problem of information acquisition and modeling it properly. However, quantization of subjective expert opinions has not yet received considerable attention as it deserves. Zimmermann & Zysno [11] showed the empirical and practical relevance of the theory of fuzzy sets to human decision making and developed an evaluation framework for 'creditworthiness rating' from a decision theoretic point of view. Biswas [1] introduced a fuzzy evaluation method of students' answer scripts. It did not consider how fuzzy mark can be generated effectively and it used also a very simplified procedure. Chakraborty [3] showed how an individual evaluator's linguistic responses can quantitatively be structured in terms of fuzzy sets and thus aggregated and compared towards making a decision. However, it has no mechanism for representing the knowledge of multiple experts.

Representation of the knowledge of multiple experts has long been a goal in expert systems development. The experts express most of their judgements by using linguistic terms drawn from different scales. Therefore, each one can express his/her preferences by means of linguistic terms assessed in linguistic terms sets with different granularity of uncertainty and/or semantics [6]. In this respect, we need a new tool for handling the uncertainty, vagueness and imprecision in linguistic assessments among multiple experts. In this connection, we propose an algorithmic approach for handling adjacent linguistic values defined in the same linguistic variable. For example, 'good'

and ‘very good’ in an evaluation, ‘long’ and ‘very long’ in hair length, or ‘medium’ and ‘large’ in size, etc., are generally adjacent linguistic values. The proposed approach can be used to capture the explicit relationship between adjacent linguistic values. In the proposed approach, the explicit relationship between adjacent linguistic values is achieved by using parameters. Since the proposed approach can be used to find explicitly the differences in linguistic assessments among multiple experts, it reduces vagueness (i.e., unsharp boundary) and ambiguity (i.e., multiplicity of meaning) in linguistic assessments among multiple experts, and also enables us to reach a collective linguistic assessment by adjusting the values of parameters. Thus, it provides a basis for improving the precision of knowledge acquisition in handling linguistic values.

2 An Algorithmic Approach for Handling Linguistic Values

Based on their behavioral experiment, Zwick et al [12] recommended the five good distance measures, i.e., S_4 , q_∞ , q_* , Δ_∞ , Δ_* , between fuzzy subset A and B of a universe of discourse U. We note that the five good distance measures concentrate their attention on a single value rather than performing some sort of averaging or integration. In the case of S_4 , attention focuses on the particular x-value where the membership function of $A \cap B$ is largest; in q_∞ and Δ_∞ , attention focuses on the α -level set where the x-distance is largest; in q_* and Δ_* , attention focuses on the x-distance at the highest membership grade. Considering the result of their behavioral experiment, we know that the reduction of complicated membership functions to a single ‘slice’ may be the intuitively natural way for human beings to combine and process fuzzy concepts. Moreover, we know that the relationship between two fuzzy subsets can be efficiently represented by a limited number of features. From these ideas, we propose an algorithmic approach for handling linguistic values (fuzzy subsets).

Zadeh [10] described that granulation of an object A leads to a collection of granules of A, with a granule being a clump of points (objects) drawn together by indistinguishability, similarity or functionality. Granulation involves decomposition of whole into parts. Modes of information granulation in which the granules are crisp play important roles in a wide variety of methods, approaches and techniques. Important though it is, crisp information granulation has a major blind spot. More specifically, it fails to reflect the fact that in much, perhaps most, of human reasoning and concept formation the granules are fuzzy rather than crisp. In the case of a human body, for example, the granules are fuzzy in the sense that the boundaries of the head, neck, arms, legs, etc., are not sharply defined. Furthermore, the granules are associated with fuzzy attributes, e.g., length, color, and texture in the case of hair. In turn, granule attributes have fuzzy values, e.g., in the case of the fuzzy attribute length (hair), the fuzzy values might be ‘short’, ‘long’, ‘very long’, etc. The fuzziness of granules, their attributes and their values is characteristic of the ways in which human concepts are formed, organized and manipulated. Fuzzy set may be used to design intelligent systems on the basis of knowledge expressed in natural language. Many aspects of different activities in the real world can not be assessed in a quantitative form, but rather in a qualitative one (i.e., with vague or imprecise knowledge). In these cases, a better approach may be to use linguistic assessments instead of

numerical values. The fuzzy linguistic approach represents qualitative aspects as linguistic values by means of linguistic variables [9]. Fuzzy sets are a means for mathematical modeling of natural language semantics. The linguistic values (terms) can be represented by membership functions. The fuzzy linguistic approach has been successfully applied to different areas such as decision making [5], information retrieval [2], cognition [4, 7], aggregation [8], etc.

The experts express most of their judgements by using linguistic terms drawn from different scales. Therefore, each one can express his/her preferences by means of linguistic terms assessed in linguistic terms sets with different granularity of uncertainty and/or semantics [6]. In this respect, we need a new tool for handling the uncertainty, vagueness and imprecision in linguistic assessments among multiple experts. In this connection, we propose an algorithmic approach for handling adjacent linguistic values defined in the same linguistic variable. It can easily capture the explicit relationship between adjacent linguistic values.

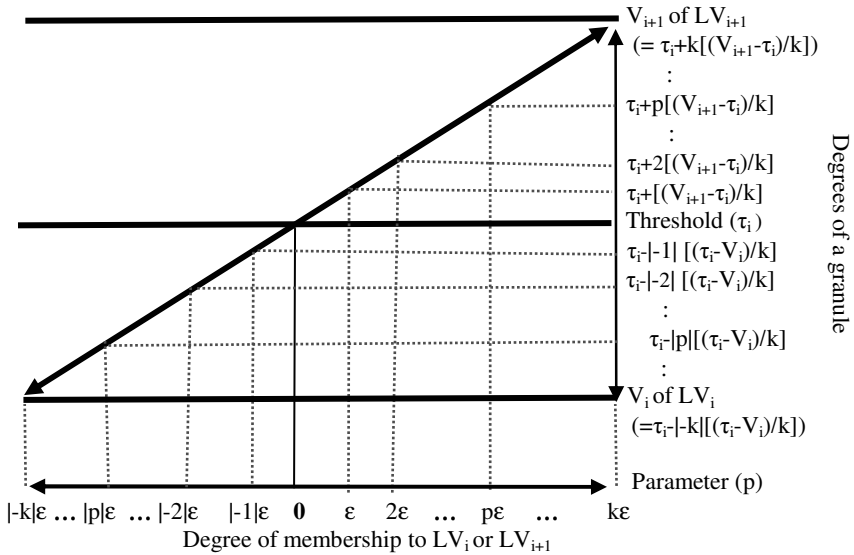


Fig. 1. Handling adjacent linguistic values

In Fig. 1, let LV_i and LV_{i+1} denote adjacent linguistic values. For example, two linguistic values ‘good’ (LV_i) and ‘very good’ (LV_{i+1}) in an evaluation are generally adjacent linguistic values. We assume that the values V_i and V_{i+1} are included in the kernel (or the full membership) of fuzzy subsets for adjacent linguistic values LV_i and LV_{i+1} , respectively. Threshold (τ_i) (an interval $[I_i, I_{i+1}]$, where $I_i \leq I_{i+1}$) may be regarded as a non-deterministic region between LV_i and LV_{i+1} . We note that the fuzzier the information regarding a linguistic assessment, the larger the interval $[I_i, I_{i+1}]$. The values of $V_i, V_{i+1}, I_i,$ and I_{i+1} are differently determined depending on individuals’ subjectivity with respect to the adjacent linguistic values.

Remark 1. In Fig. 1, the ε divides the interval $[-1, 1]$ into $[-k\varepsilon (= -1), \dots, -2\varepsilon, -\varepsilon, \mathbf{0}, \varepsilon, 2\varepsilon, \dots, k\varepsilon (=1)]$, where $0 < \varepsilon \leq 1$, and $k = (1/\varepsilon)$. For instance, the relationship between k and ε is determined by using the formula : $k = 10^n$, if $\varepsilon = 10^{-n}$, n is a positive integer.

Remark 2. In Fig. 1, the interval $[V_i, V_{i+1}]$ is divided into $(2k+1)$ levels as follows : $\tau_i - k[(\tau_i - V_i)/k] (=V_i)$, $\tau_i - (k-1)[(\tau_i - V_i)/k]$, \dots , $\tau_i - 2[(\tau_i - V_i)/k]$, $\tau_i - [(\tau_i - V_i)/k]$, τ_i , $\tau_i + [(V_{i+1} - \tau_i)/k]$, $\tau_i + 2[(V_{i+1} - \tau_i)/k]$, \dots , $\tau_i + (k-1)[(V_{i+1} - \tau_i)/k]$, $\tau_i + k[(V_{i+1} - \tau_i)/k] (=V_{i+1})$. We assume that a given granule (g_i) is located between V_i and V_{i+1} , (i.e., $V_i \leq g_i \leq V_{i+1}$). It should be noted that linguistic values are largely divided into two types of linguistic values, i.e., monotonically nondecreasing linguistic values or monotonically nonincreasing linguistic values. Monotonically nondecreasing linguistic values have a property that they converge to the V_{i+1} as a given granule (g_i) increases, as shown in Fig. 1. On the contrary, monotonically nonincreasing linguistic values have a property that they converge to the V_i as a given granule (g_i) increases. More specifically, for monotonically nondecreasing linguistic values, consider a linguistic variable ‘popularity’ on national parks. We assumed that it is linguistically assessed based on the number of visitors. Let LV_i and LV_{i+1} denote adjacent linguistic values ‘popular’ and ‘very popular’ and let their values V_i and V_{i+1} be the number of visitors, respectively. Then it converges to the V_{i+1} as a given granule (g_i) increases. In this case, the given granule (g_i) is the number of visitors. On the contrary, for monotonically nonincreasing linguistic values, let LV_i and LV_{i+1} denote adjacent linguistic values ‘unpopular’ and ‘very unpopular’ and let their values V_i and V_{i+1} be the number of visitors, respectively. Then it converges to the V_i as a given granule (g_i) increases. In this paper, we will consider monotonically nondecreasing linguistic values such as ‘popular’, ‘big’, etc., as in Fig. 1. Monotonically nonincreasing linguistic values such as ‘unpopular’, ‘small’, etc., can be handled in the reverse way.

Remark 3. In Fig. 1, the degree of granule per level to the linguistic value $LV_i(D_{dgl})$ is $[(\tau_i - V_i)/k]$, and the degree of granule per level to the linguistic value $LV_{i+1}(U_{dgl})$ is $[(V_{i+1} - \tau_i)/k]$.

Remark 4. The connecting parameter, p , representing the relationship between a given granule (g_i) and a degree of membership to a linguistic value, is computed as follows : (i) when a given granule (g_i) > threshold (τ_i) (i.e., *upward granularity* to a linguistic value LV_{i+1} occurs), $p = (g_i - \tau_i)/(U_{dgl})$, or (ii) when a given granule (g_i) < threshold (τ_i) (i.e., *downward granularity* to a linguistic value LV_i occurs), $p = (g_i - \tau_i)/(D_{dgl})$, where $V_i \leq g_i \leq V_{i+1}$.

Definition 1. Based on Remarks 1-4, degrees of a granule (i.e., perceptions of distance, size, color, speed, etc.) can be parameterized with $(2k+1)$ levels.

Case 1 : *Upward granularity* (U_g) to a linguistic value LV_{i+1} occurs when a given granule (g_i) > threshold (τ_i). *Upward granularity* can be parameterized with k levels, where $0 < p \leq k$. (k levels occur)

Case 2 : A non-deterministic situation occurs when a given granule (g_i) = threshold (τ_i). In this case, $p = 0$, (i.e., threshold (τ_i) in Fig. 1). (1 level occurs)

Case 3 : *Downward granularity* (D_g) to a linguistic value LV_i occurs when a given granule (g_i) < threshold (τ_i). *Downward granularity* can be parameterized with k levels, where $-k \leq p < 0$. (k levels occur)

Definition 2. Upward granularity to a linguistic value LV_{i+1} (i.e., $\tau_i+p(U_{dgl})$) and downward granularity to a linguistic value LV_i (i.e., $\tau_i-|p|(D_{dgl})$) indicate the perception on a given granule (g_i). In Fig. 1, we note that the parameter, p , may be regarded as a connecting link between a given granule and a degree of membership to a linguistic value. The parameter, p , is a real number in $[-k, k]$, where k is the number of levels to the values V_i or V_{i+1} which is included in the kernel of fuzzy subsets for adjacent linguistic values LV_i or LV_{i+1} , respectively. The degree of membership to a linguistic value LV_i or LV_{i+1} is determined according to the value of the connecting parameter p . That is, the degree of membership to a linguistic value LV_i or LV_{i+1} at the downward granularity (i.e., $\tau_i-|p|(D_{dgl})$) or upward granularity (i.e., $\tau_i+p(U_{dgl})$) is $|p|\epsilon$ or $p\epsilon$, respectively, as in Fig. 1.

Zadeh [10] described that the effectiveness and successes of fuzzy logic in dealing with real-world problems rest in large measure on the use of the machinery of fuzzy information granulation. In this connection, what should be underscored is that when we talk about fuzzy information granulation we are not talking about a single fuzzy granule; we are talking about a collection of fuzzy granules which result from granulating a crisp or fuzzy object. One of the most basic facets of human cognition relates to the perception of dependencies and relations. In this perspective, we develop an algorithmic approach representing the relationship between adjacent linguistic values. Based on Remarks 1-4, Definitions 1-2, the proposed approach is executed by the following algorithm :

Algorithm 1

/* Let $k\epsilon = 1$, where $0 < \epsilon \leq 1$ (see Remark 1). The parameter, p , is a connecting parameter between a given granule and a degree of membership to a linguistic value, and it is a real number in $[-k, k]$. */

Step 1 : Get a given granule (g_i).

Step 2 : Determine the adjacent linguistic values (i.e., LV_i, LV_{i+1}), where $V_i \leq g_i \leq V_{i+1}$.

Step 3 : Determine the classification parameters (i.e., V_i, V_{i+1}, τ_i).

Step 4 : Determine the preciseness parameter (i.e., ϵ).

Step 5 : Compare a given granule (g_i) with threshold (τ_i).

Case $g_i > \tau_i$: /* *Upward granularity* to a linguistic value LV_{i+1} occurs */

- Compute *upward degree of granule per level* (U_{dgl}) to the linguistic value LV_{i+1} :

$U_{dgl} = [(V_{i+1}-\tau_i)/k]$ (see Remark 3).

- Compute *the connecting parameter* (p) : $p = (g_i-\tau_i)/(U_{dgl})$.
- Determine *upward granularity*(U_g) to a linguistic value LV_{i+1} : $U_g = \tau_i + p(U_{dgl})$.
- Determine *a degree of membership* to a linguistic value LV_{i+1} at U_g : $\mu_{LV_{i+1}}(U_g) = p\epsilon$.

Case $g_i = \tau_i$: Non-deterministic situation occurs, (i.e., the threshold (τ_i) in Fig. 1). Let $p = 0$. In this case, a given granule is linguistically assessed as '*between LV_i and LV_{i+1}* '.

Case $g_i < \tau_i$: /* *Downward granularity* to a linguistic value LV_i occurs */

- Compute *downward degree of granule per level* (D_{dgl}) to the linguistic value LV_i : $D_{dgl} = [(\tau_i - V_i)/k]$ (see Remark 3).
- Compute *the connecting parameter* (p): $p = (g_i - \tau_i)/(D_{dgl})$.
- Determine *downward granularity*(D_g) to a linguistic value LV_i : $D_g = \tau_i - |p| (D_{dgl})$.
- Determine *a degree of membership* to a linguistic value LV_i at D_g : $\mu_{LV_i}(D_g) = |p| \epsilon$.

The proposed approach shows the following properties : First, monotonicity is generally comfortable to human reasoning and concept formation. The proposed approach is designed based on the monotonicity as shown in Fig. 1 and Algorithm 1. Second, the proposed approach can be used to capture the explicit relationship between adjacent linguistic values. Third, degree of membership to a linguistic value is, monotonically and adaptively, changed according to a degree of granule. In addition, it converges to the full membership of the linguistic values LV_i or LV_{i+1} as a given granule (g_i) decreases or increases, respectively. Fourth, we can change the number of levels to the linguistic values LV_i or LV_{i+1} by adjusting the value of preciseness parameter ϵ , where $0 < \epsilon \leq 1$. Fifth, if we use a very small ϵ value (i.e., $\epsilon \rightarrow 0$), we can handle the continuous granularity. Thus, the proposed approach can be used both a crisp object (discrete granularity) and fuzzy object (continuous granularity). For example, the number of apples, the number of visitors, etc., are discrete granularity, whereas color of hair, driving a car, etc., are continuous granularity. Sixth, the proposed approach can be similarly extended to multiple linguistic values defined in the same linguistic variable. That is, if we define the multiple values of classification parameters (i.e., V_i, V_{i+1} and τ_i) and preciseness parameter ϵ_i , where $1 \leq i \leq n-1$, regarding the multiple linguistic values (i.e., $LV_1, LV_2, \dots, LV_{n-1}, LV_n$), the linguistic assessment with a degree of membership to a linguistic value can be systematically computed according to a given granule. It is a further step toward an automatic perception handling. We summarize the differences between the proposed approach and existing approaches in Table 1.

Table 1. Comparisons between the proposed approach and existing approaches

| Attributes | The proposed approach | Existing approaches |
|-------------------------------|-----------------------|---------------------|
| Handling linguistic values | Algorithmic | Ad-hoc |
| A degree of membership | Granulation | Ad-hoc |
| Automatic perception handling | Easy | Difficult |

3 Collective Linguistic Assessment Among Multiple Experts

Representation of the knowledge of multiple experts has long been a goal in expert systems development [4]. As aforementioned, the proposed approach can be used to capture the explicit relationship between adjacent linguistic values. In the proposed approach, the explicit relationship between adjacent linguistic values is achieved by using parameters. In Fig. 1, using the values of classification parameters (i.e., V_i, V_{i+1} and τ_i) and preciseness parameter ϵ , we can explicitly represent the relationship between adjacent linguistic values. In this respect, the proposed approach can be

employed as a useful tool for discovering hidden relationship between adjacent linguistic values. Using the proposed approach, we can explicitly capture the differences of individuals' subjectivity with respect to the adjacent linguistic values. In this connection, the proposed approach can be used to find explicitly the differences in linguistic assessments among multiple experts, thus, ensuring that all are speaking 'the same language'. Thus, it enables us to reach a collective linguistic assessment by adjusting the values of parameters (i.e., V_i , V_{i+1} , τ_i , and ϵ). In Fig. 2 (see Appendix), we show the procedure for the collective linguistic assessment among multiple experts.

Example 1. Consider a linguistic variable 'popularity' on national parks. We assumed that it is linguistically assessed based on the number of visitors. Let a given granule (g_i) be 8630 visitors/day. If the given granule is determined between 'popular' and 'very popular' by multiple experts, in this case, adjacent linguistic values LV_i and LV_{i+1} become 'popular' and 'very popular', respectively. Assume that the classification parameters V_i , V_{i+1} , and threshold (τ_i) are determined by multiple experts, as 7,000/day, 10,000/day and $[I_i, I_{i+1}] = [8400, 8600]$, respectively. If we apply $\epsilon = 10^{-2}$ to the proposed approach, then $k = 10^2$ as in Remark 1. In this case, since the given granule (i.e., 8630) is greater than threshold (τ_i), (i.e., 8600), where $\tau_i = I_{i+1}$, *upward granularity* is occurred. The degree of granule per level to the linguistic value LV_{i+1} becomes $(V_{i+1}-\tau_i)/k = (10000-8600)/10^2 = 14$. Now, compute the connecting parameter $p = [g_i-\tau_i]/[(V_{i+1}-\tau_i)/k] = [8630-8600]/[(10000-8600)/10^2] = 30/14 \cong 2.14$. In this case, the given granule (g_i) is located between level 2 (i.e., 8628) and level 3 (i.e., 8642). As a result, a *degree of membership* to LV_{i+1} (i.e., 'very popular') becomes $p\epsilon = 2.14 (10^{-2}) = 2.14/100$. Similarly, if the given granule (g_i) is 8370, *downward granularity* is occurred because the given granule (i.e., 8370) is less than threshold (τ_i), (i.e., 8400), where $\tau_i = I_i$. Compute the connecting parameter $p = [g_i-\tau_i]/[(\tau_i-V_i)/k] = [8370-8400]/[(8400-7000)/10^2] = -30/14 \cong -2.14$. As a result, a *degree of membership* to LV_i (i.e., 'popular') becomes $|p| \epsilon = 2.14 (10^{-2}) = 2.14/100$.

In Example 1, if we use a smaller ϵ value, we can obtain a more sophisticated collective linguistic assessment among multiple experts.

4 Conclusion

Based on the behavioral experiment by Zwick et al [12], an algorithmic approach for handling linguistic values is proposed. The proposed approach may be employed as a useful tool for discovering hidden relationship between linguistic values. In this connection, it enables us to reach a collective linguistic assessment among multiple experts. It provides a basis for improving the precision of knowledge acquisition in handling linguistic values.

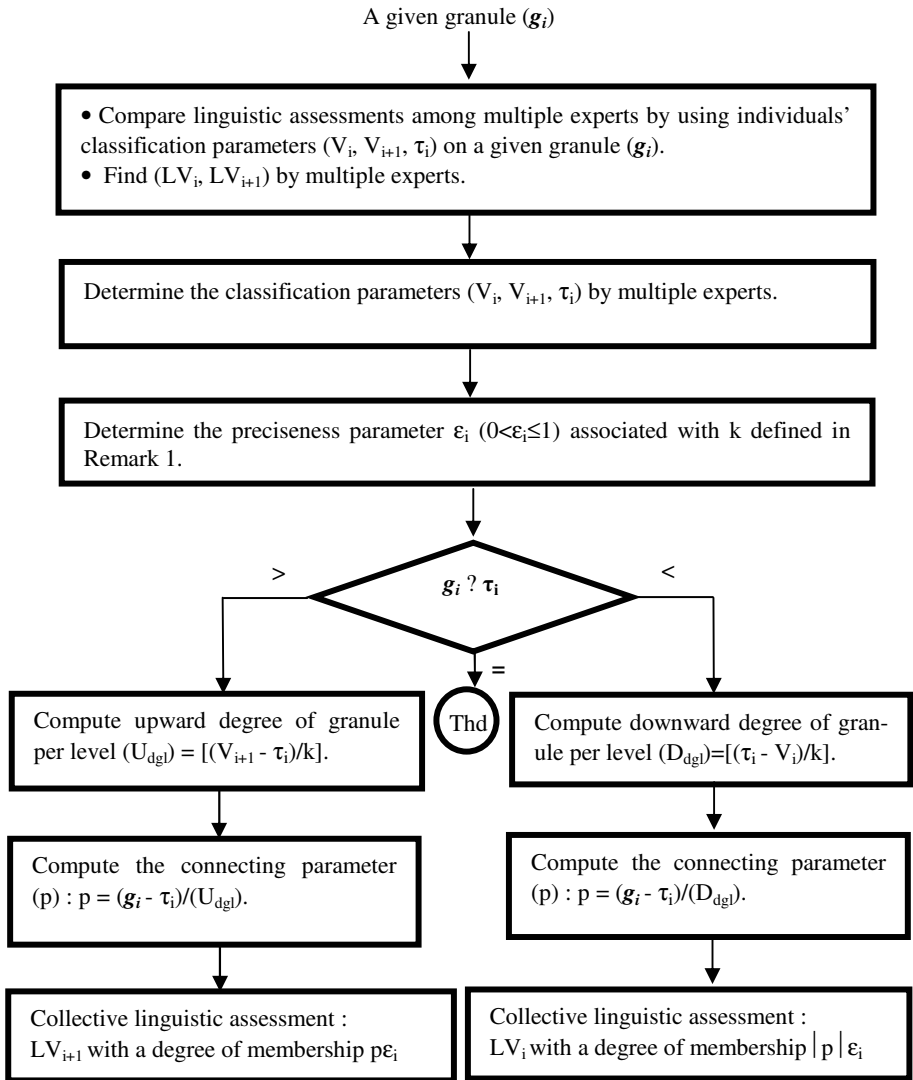
In the meantime, we show that the degree of membership to a linguistic value is, monotonically and adaptively, changed according to a degree of granule. We also show that the linguistic assessment with a degree of membership to a linguistic value can be systematically computed according to a given granule. It is a further step toward an automatic perception handling.

Acknowledgement. The author wishes to thank Prof. L. A. Zadeh, University of California, Berkeley, for his inspirational address on the perceptual aspects of humans in the BISC (Berkeley Initiative in Soft Computing) seminars and group meetings.

References

1. Biswas, R.: An Application of Fuzzy Sets in Students' Evaluation, *Fuzzy Sets and Systems* 74 (1995) 187-194.
2. Bordogna, G., Passi, G.: A Fuzzy Linguistic Approach Generalizing Boolean Information Retrieval : A Model and Its Evaluation, *J. Amer. Soc. Inf. Sci.* 44 (1993) 70-82.
3. Chakraborty, D.: Structural Quantization of Vagueness in Linguistic Expert Opinions in an Evaluation Programme, *Fuzzy Sets and Systems* 119 (2001) 171-186.
4. Cole, J. R., Persichitte, K. A.: Fuzzy Cognitive Mapping : Applications in Education, *Int. J. Intelligent Systems* 15 (2000) 1-25.
5. Delgado, M., Verdegay, J. L., Vila, M. A.: Linguistic Decision Making Models, *Int. J. Intelligent Systems* 7 (1992) 479-492.
6. Herrera, F., Martinez, L.: A Model Based on Linguistic 2-Tuples for Dealing with Multi-granular Hierarchical Linguistic Contexts in Multi-Expert Decision-Making, *IEEE Trans. on SMC (part B)* 31 (2001) 227-234.
7. Kosko, B.: Fuzzy Cognitive Maps, *Int. J. Man-Machine Studies* 24 (1986) 65-75.
8. Peláez, J. I., Doña, J. M. : LAMA: A linguistic aggregation of majority additive operator, *International Journal of Intelligent Systems* 18 (2003) 809-820.
9. Zadeh, L. A.: The Concept of a Linguistic Variable and Its Applications to Approximate Reasoning (part I), *Inf. Sci.* (8) (1975) 199-249.
10. Zadeh, L. A.: Toward a Theory of Fuzzy Information Granulation and Its Centrality in Human Reasoning and Fuzzy Logic, *Fuzzy Sets and Systems* 90 (1997) 111-127.
11. Zimmermann, H. J., Zysno, P.: Decisions and Evaluations by Hierarchical Aggregation of Information, *Fuzzy Sets and Systems* 10 (1983) 243-260.
12. Zwick, R., Carlstein, E., Budescu, D. V.: Measures of Similarity among Fuzzy Concepts : A Comparative Analysis, *Int. J. of Approximate Reasoning*, Vol. 1 (1987) 221-242

Appendix



* Thd (Threshold) : it is linguistically assessed as 'between LV_i and LV_{i+1} '.

Fig. 2. Collective linguistic assessment among multiple experts

An IA Based Approach for the Optimal Design of Traffic-Monitor Systems

Yi-Chih Hsieh¹, Yung-Cheng Lee², and Ta-Cheng Chen³

¹ Department of Industrial Management, National Formosa University
Huwei, Yunlin 632, Taiwan
yhsieh@nfu.edu.tw

² Department of Electrical Engineering, WuFeng Institute of Technology
Ming-Hsiung, Chia-Yi 621, Taiwan
yclee@mail.wfc.edu.tw

³ Department of Information Management, National Formosa University
Huwei, Yunlin 632, Taiwan
tchen@nfu.edu.tw

Abstract. To improve the safety of drivers and walkers in a city, several traffic monitors are usually set on lanes. These traffic monitors can also improve the security of communities. In this paper, we integrate the so-called linear/circular consecutive- k -out-of- n :F systems into our proposed traffic-monitor system. The objective is to find the optimal design of monitors under limited budget for the system. The main purposes of this paper are : (1) to propose a new traffic-monitor system, (2) to present an immune algorithm (IA) for the optimal design of traffic monitors, and (3) to report numerical results of various parameters by the proposed algorithm. It is shown that the proposed immune algorithm performs well for all test problems.

1 Introduction

With the increase of cars and walkers, traffic monitors are usually set on specific heavy-traffic lanes and circles to enhance the safety of drivers and walkers. Monitors are operative 24 hours per day. The films by the monitors would be useful to judge the responsibility of drivers and walkers for insurance companies and to improve the security of communities. For a specific lane L , there are usually several monitors set in equal distance as those of Fig 1.

In Fig 1, there are n traffic monitors and each monitor can film equal width on lane L . Monitors 1 and 2 are responsible for line segment $[a,b]$ simultaneously, and this line segment fails if and only if both monitors 1 and 2 are failed. Similarly, line segment $[b,c]$ fails if and only if both monitors 2 and 3 are failed,. Thus, the failure reliability of traffic-monitor system for lane L is defined as the probability that there is one or more line segments failed.

Note that the traffic-monitor system in Fig 1 is a so-called consecutive-2-out-of- n :F system (or C(2, n :F) system), in which there are n linearly connected

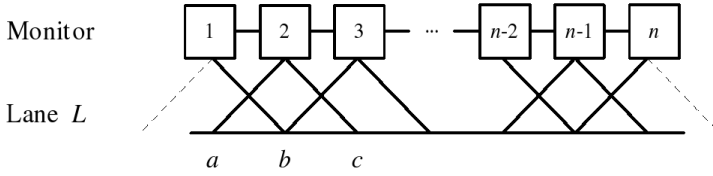


Fig. 1. Linear lane with n traffic monitors

components, and it fails if and only if there are consecutive 2 or more than 2 components failed. The general $C(k,n:F)$ system consists of n linearly connected components, and it fails if and only if there are consecutive k or more than k components failed (Fig 2). Such a $C(k,n:F)$ system has caught much attention since 1980 due to the followings (Chao et al. (1995)).

(1) $C(k,n:F)$ system usually has much higher probability than the series systems.

(2) $C(k,n:F)$ system is less expensive than the parallel systems.

Since 1980, much research has been devoted to the derivation of exact formulae for the reliability of $C(k,n:F)$ system. In the early years, most of the proposed formulae were based upon the recursive equations and assumed that all the components are independent and have the same probability. In 1984, Chao and Lin (1984) first observed that the general $C(k,n:F)$ system can be imbedded in a Markov chain with 2^k states. However, only systems with small k can be manipulated. Later, Fu (1986) successfully reduced the Markov chain into $k+1$ states and considerably simplified the probability structure of $C(k,n:F)$ system.

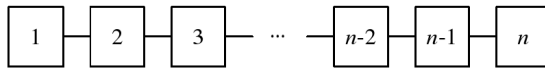


Fig. 2. Linear $C(k,n:F)$ system

The circular $C(k,n:F)$ system is similar to linear $C(k,n:F)$ system. The system consists of n components in a circle and it fails if and only if there are consecutive k or more than k components failed. For example, the system in Fig 3 is a circular $C(2,8:F)$ system (i.e., $k=2$), and if both components 1 and 2 are failed will lead to system failure.

In this paper, we will combine the linear and circular $C(k,n:F)$ systems into a traffic-monitor system, in which the typical $C(k,n:F)$ system systems are special cases of the proposed system. Additionally, we will investigate the optimal design of the traffic-monitor system whose objective is to minimize the system failure reliability subject to the budget constraint. In this paper, a new immune algorithm is presented to solve such optimization problems. Numerical results of various combinations of parameters are reported and discussed.

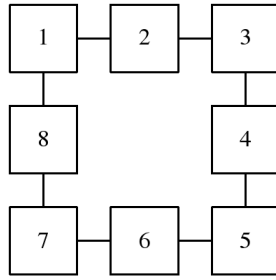


Fig. 3. Circular $C(2,8:F)$ system

2 The New Proposed Traffic-Monitor Systems

For convenience, we consider the following general traffic-monitor system in Fig 4 as an example. Fig 4(a) shows the traffic lanes and a circle (from A to P) in a city, and Fig 4(b) shows the 16 possible locations for traffic monitors. Monitors 1 and 2 are responsible for lane A , monitors 2 and 3 are responsible for lane B , and monitors 3 and 6 are responsible for lane E etc. Note that when both monitors 1 and 2 are failed, lane A cannot be monitored and it is defined to be failed.

Similarly, lane B is failed if and only if monitors 2 and 3 are failed. Therefore, there are 16 minimal cuts for the system, namely, $C_A=\{1,2\}$, $C_B=\{2,3\}$, $C_C=\{2,4\}$, $C_D=\{2,5\}$, $C_E=\{3,6\}$, $C_F=\{6,7\}$, $C_G=\{6,11\}$, $C_H=\{7,8\}$, $C_I=\{7,9\}$, $C_J=\{9,10\}$, $C_K=\{9,14\}$, $C_L=\{10,11\}$, $C_M=\{10,13\}$, $C_N=\{11,12\}$, $C_O=\{14,15\}$ and $C_P=\{14,16\}$. The traffic-monitor system fails if and only if there is one or more minimal cuts of monitors failed.

Suppose that for each possible location, we have four types of monitors to choose, namely, 3 (high quality), 2 (medium quality), 1 (low quality), 0 (no monitor). The higher quality of monitor will have higher reliability and higher cost. Therefore, the optimal design problem is to find the optimal assignments of monitors for these 16 possible locations, and there are $4^{16} (=4.29 \times 10^9)$ combinations for this example.

Note that:

- 1.The traffic-monitor system contains both linear $C(k,n:F)$ system and circular $C(k,n:F)$ system.

- 2.The well known recursive methods for system reliability of the typical linear $C(k,n:F)$ system and circular $C(k,n:F)$ system can not be used for the new proposed traffic-monitor system.

2.1 System Reliability of Traffic-Monitor System

Suppose that $C_1, C_2, C_3, \dots, C_s$ are s minimal cuts for a specific traffic-monitor system. The reliability of the system can be obtained by the inclusion-exclusion principle as:

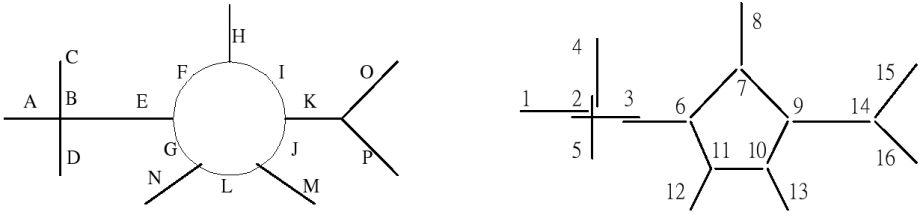


Fig. 4. The general traffic-monitor system

$$\begin{aligned}
 R_{sys} &= 1 - P(C_1 \cup C_2 \cup \dots \cup C_s) \\
 &= 1 - \left\{ \begin{aligned} &\sum_i P(C_i) - \sum_{i < j} P(C_i C_j) + \sum_{i < j < k} P(C_i C_j C_k) - \dots \\ &+ (-1)^{(s+1)} P(C_1 C_2 \dots C_s) \end{aligned} \right\} \quad (1)
 \end{aligned}$$

Note that there are 2^s terms in (1) and it is time consuming to compute R_{sys} when s is larger. In this paper, we will use the following disjoint-subset method to compute the system reliability $R_{sys}=1-P(C_1 \cup C_2 \cup \dots \cup C_s)$. Consider the following terms.

$$P(C_1) \tag{2}$$

$$P(C_2) - P(C_2 C_1) \tag{3}$$

$$P(C_3) - P(C_3(C_2 \cup C_1)) \dots \tag{4}$$

$$P(C_s) - P(C_s(C_1 \cup C_2 \dots \cup C_{s-1})) \tag{5}$$

Thus $P(C_1 \cup C_2 \cup \dots \cup C_s)$ is the sum of terms from (2) to (5). There are several advantages for this disjoint-subset method.

1.The method may cancel redundant terms in advance.

2.The method is more “efficient” than inclusion-exclusion principle in computational complexity (Hsieh (2006)).

3.The method is more “flexible” than inclusion-exclusion principle (Hsieh (2006)). For example, if we add a new minimal cut C_{s+1} into the s minimal cuts, this disjoint-subset method only has to compute:

$$P(C_{s+1}) - P(C_{s+1}(C_1 \cup C_2 \dots \cup C_s)) \tag{6}$$

and $P(C_1 \cup C_2 \cup \dots \cup C_{s+1})$ is the sum of terms from (2) to (6).

2.2 Optimization of Design

In the past decades, the design of systems/components is always one of the main issues of research (Kuo and Prasad (2000)). As known, the optimal design of typical $C(k,n:F)$ system has been investigated by several researchers. For example, Bai et al (1991), Du and Hwang (1990), Zuo and Kuo (1990) etc. However, only linear or circular $C(k,n:F)$ systems were studied separately. Interested readers are referred to the excellent survey paper by Kuo and Prasad (2000).

Next, we will define the optimal design problems and notations for our proposed traffic-monitor system. Note that:(i) our traffic-monitor system contains linear and circular $C(k,n:F)$ systems simultaneously, and (ii) there are multiple types of monitors in our traffic-monitor system.

Assumptions.

1. There are two possible states for the system, i.e., failed and operative.
2. Monitor i has t_i types, $\{0,1,\dots,t_i-1\}$, where 0 denotes no monitor, $1 \leq i \leq n$.
3. The reliabilities and costs of monitors are known.
4. All monitors in the system are independent.
5. Any failure of minimal cut of monitors will incur the system failure.

Notations.

S :the monitor assignment set.

n :the number of locations of monitors in the system.

p_{ij} :the reliability of monitor i under type j , $1 \leq i \leq n, j \in T_i=\{0,1,\dots,t_i-1\}$.

c_{ij} :the cost of monitor i under type j , $1 \leq i \leq n, j \in T_i=\{0,1,\dots,t_i-1\}$.

B :the total budget.

$P_F(S)$:the system failure probability under S .

The nonlinear mathematical programming for the design problem is:

$$\min_{S \in \Omega} P_F(S) \tag{7}$$

$$\text{st } \sum_{i \in S} \sum_{j \in T_i} c_{ij} \leq B, 0 \leq P_F(S) \leq 1 \tag{8}$$

where the objective is to minimize the system failure probability $P_F(S)$ subject to the budget constraint.

Since the typical maintenance problem for linear $C(k,n:F)$ system is a NP hard problem (Flynn and Chung (2002, 2004)), clearly this proposed optimal design problem for traffic-monitor system is also a NP hard problem. As known, Branch-and-Bound Method can be used to solve the proposed problem. However, it is both time and memory consuming especially when the problem size is larger (Flynn and Chung (2002)). Heuristics can be also developed to solve the proposed design problems. However, no optimal solution is guaranteed and it usually converges to local optimal solutions even for small problems (Flynn and Chung (2004)). Additionally, Enumeration Method and Genetic Algorithm can be used to solve the proposed problem. But the drawbacks of Enumeration Method are similar to those of Branch-and-Bound Method. In addition, Genetic Algorithm is sometimes very sensitive to the settings of corresponding parameters and then converges to local optimal solutions.

3 Immune Algorithm

The natural immune system of all animals is a very complex system for defense against pathogenic organisms. A two-tier line of defense is in the system including the innate immune system and the adaptive immune system. The basic

components are lymphocytes and antibodies (Farmer (1986)). The cells of the innate immune system are immediately available to combat against a wide variety of antigen without previous exposure to them. The antibody production in response to a determined infectious agent (antigen) is the adaptive immune response mediated by lymphocytes which are responsible for recognition and elimination of the pathogenic agents (De Castro and Timmis (2002)). The cells in the adaptive system are able to develop an immune memory so that they can recognize the same antigenic stimulus when it is presented to the organism again. Also, all the antibodies are produced only in response to specific infections. There are two main types of lymphocytes: B-lymphocytes (B-cells) and T-lymphocytes (C-cells). B-cells and T-cells carry surface receptor molecules capable of recognizing antigens. The B-cells produced by the bone marrow show a distinct chemical structure and can be programmed to make only one antibody that is placed on the outer surface of the lymphocyte to act as a receptor. The antigens will only bind to these receptors with which it makes a good fit (Huang (1999)).

To distinguish and eliminate the intruders of the organism is the main task of the immune system so that it must have the capability of self/nonself discrimination. As mentioned previously, various antibodies can be produced and then can recognize the specific antigens. The portion of antigen recognized by antibody is called epitope which acts as an antigen determinant. Every type of antibody has its own specific antigen determinant which is called idiotope. Moreover, in order to produce enough specific effector cells to against an infection, an activated lymphocyte has to proliferate and then differentiate into these effector cells. This process is called clonal selection (Weissman and Cooper (1993)) and followed by the genetic operations such that a large clone of plasma cell is formed. Therefore, the antibodies can be secreted and ready to bind antigens. According to above facts, Jerne (1973) proposed an idio-type network hypothesis which is based on the clonal selection theory. In his hypothesis, some types of recognizing sets are activated by some antigens and produce an antibody which will then activate other types of recognizing sets. By this way, the activation is propagated through entire network of recognizing sets via antigen-antibody reactions. It is noted that the antigen identification is not done by a single or multiple recognizing sets but by antigen-antibody interactions. The more details are referred to Huang (1999, 2000). From this point of view, for solving the combinatorial optimization problems, the antibody and antigen can be looked as the solution and objection function respectively.

3.1 The Procedure of Immune Algorithm

The steps of proposed immune algorithm are as follows:

- Step 1.** Generate an initial population of strings (antibodies) randomly.
- Step 2.** Evaluate each individual in current population and calculate the corresponding fitness value for each individual.
- Step 3.** Select the best n individuals with highest fitness values.

- Step 4.** Clone the best n individuals (antibodies) selected in Step 3. Note that the clone size for each select individual is an increasing function of the affinity with the antigen.
- Step 5.** The set of the clones in Step 4 will suffer the genetic operation process, i.e., crossover and mutation (Michalewicz (1994)).
- Step 6.** Calculate the new fitness values of these new individuals (antibodies) from Step 5. Select those individuals who are superior to the individuals in the memory set, and then the superior individuals replace the inferior individuals in the memory set. While the memory set is updated, the individuals will be eliminated while their structures are too similar.
- Step 7.** Check the stopping criterion, if not stop then go to Step 2. Otherwise go to next step.
- Step 8.** Stop. The optimal or near optimal solution(s) can be obtained from the memory set.

3.2 The Representation Mechanism

In our implementation, the integer solutions are represented by strings of binary digits. Each string consisting of substring denotes the types of monitors. In the above procedures, the clonal selection and affinity maturation processes are described in details by De Castro and Von Zuben (2000). The stopping criterion is the maximum iterations in this article.

Because of the soul of diversity in the IAs, the quality of solutions in the feasible space can be better guaranteed and obtained. So, a suppression process (diversity embodiment) is needed and shown on the Step 6 in the proposed IAs procedure. In this study, for each antibody represented by a binary string can be translated into an integer string which illustrates the types of monitors. The diversity in each pair of antibody i (Ab_i) and antibody j (Ab_j) can be evaluated by calculating their affinity (f_{ij}) by following way:

$$f_{ij} = ||Ab_i - Ab_j|| . \quad (9)$$

While the affinity between each pair of antibodies in memory is obtained, the antibodies will be eliminated if the affinity is less than the predefined threshold. So, the diversity of the antibodies in memory is embodied.

4 Numerical Results and Discussions

To test the proposed immune algorithm for the traffic-monitor system, we set $T_i = \{0, 1, 2, 3\}$ for all i and let:

(1) The available budgets B are various from 200 to 800.

(2) Case I: (medium quality of monitors) $p_{i0} = 0$, $p_{i1} = 0.85$, $p_{i2} = 0.90$, $p_{i3} = 0.95$, $c_{i0} = 0$, $c_{i1} = 10$, $c_{i2} = 20$, and $c_{i3} = 60$.

(3) Case II: (high quality of monitors) $p_{i0} = 0$, $p_{i1} = 0.90$, $p_{i2} = 0.95$, $p_{i3} = 0.98$, $c_{i0} = 0$, $c_{i1} = 10$, $c_{i2} = 20$, and $c_{i3} = 60$.

Table 1. Numerical results of top-3 solutions for system in Fig 4

| B | Immune Algorithm | | | | | |
|-----|---------------------------------|-----------------------------------|-----------------------------|---------------------------------|---------------------------------|-----------------------------|
| | Case I | | | Case II | | |
| | Rsys (1- P _F (S)) | Top-3 Assignments | ITER* CPU** Total CPU | Rsys (1- P _F (S)) | Top-3 Assignments | ITER* CPU** Total CPU |
| 200 | 0.801152 | 1 2 1 1 1 1 2 1 1 1 2 1 1 2 1 1 | 91 | 0.918918 | 1 2 1 1 1 1 2 1 1 1 2 1 1 2 1 1 | 68 |
| | 0.799694 | 1 2 1 1 1 1 1 1 2 2 1 1 2 1 1 | 2050.157 | 0.918710 | 1 2 1 1 1 1 2 1 1 1 2 1 1 2 1 1 | 1587.203 |
| | 0.797070 | 1 2 1 1 1 1 2 1 2 1 1 1 1 2 1 1 | 2259.954 | 0.918553 | 1 2 1 1 1 2 1 1 2 1 1 1 2 1 1 | 2292.843 |
| 300 | 0.868353 | 1 3 2 1 1 2 2 1 2 2 2 2 2 1 1 | 25 | 0.959896 | 1 3 2 1 1 2 2 1 2 2 2 2 2 1 1 | 69 |
| | 0.868344 | 1 3 2 1 1 2 2 2 2 2 2 1 2 2 1 1 | 580.000 | 0.959895 | 1 3 2 1 1 2 2 2 2 2 2 1 2 2 1 1 | 1442.328 |
| | 0.868097 | 1 3 2 1 1 2 2 1 2 2 2 1 2 2 2 1 | 2200.375 | 0.959775 | 1 3 2 1 1 2 2 1 2 2 2 2 1 2 1 2 | 2116.297 |
| 400 | 0.900304 | 1 3 2 1 1 2 2 2 2 2 3 2 2 3 1 2 | 50 | 0.973505 | 1 3 2 1 1 2 2 2 2 2 3 2 2 3 2 1 | 48 |
| | 0.900304 | 1 3 2 1 1 2 2 2 2 2 3 2 2 3 2 1 | 1059.547 | 0.973501 | 1 3 2 1 1 2 2 2 2 2 3 1 2 3 2 2 | 1001.297 |
| | 0.900269 | 1 3 2 1 1 2 2 2 2 2 3 1 2 3 2 2 | 2127.031 | 0.973421 | 1 3 2 2 1 2 2 2 2 2 3 2 2 3 1 1 | 2117.953 |
| 500 | 0.926279 | 1 3 2 1 1 3 3 1 2 3 3 2 2 3 1 1 | 29 | 0.982500 | 2 3 2 1 1 2 3 2 2 3 3 2 2 3 2 2 | 36 |
| | 0.926278 | 1 3 2 1 1 3 3 2 2 3 3 2 1 3 1 1 | 638.094 | 0.982500 | 1 3 2 1 2 3 2 2 3 3 2 2 3 2 2 | 758.343 |
| | 0.926061 | 1 3 2 1 1 3 3 1 2 3 3 2 1 3 1 2 | 2211.312 | 0.982500 | 2 3 2 1 1 2 3 2 2 3 3 2 2 3 2 2 | 2156.515 |
| 600 | 0.942814 | 2 3 2 2 2 3 3 2 3 3 3 2 2 3 2 2 | 16 | 0.988254 | 2 3 2 2 2 3 3 2 3 3 3 2 2 3 2 2 | 28 |
| | 0.941058 | 1 3 2 2 2 3 3 2 3 3 3 2 2 3 2 2 | 355.907 | 0.987711 | 2 3 3 2 2 3 3 2 2 3 3 2 2 3 2 2 | 636.594 |
| | 0.941058 | 2 3 2 2 1 3 3 2 3 3 3 2 2 3 2 2 | 2200.797 | 0.987403 | 1 3 2 2 2 3 3 2 3 3 3 2 2 3 2 2 | 2351.438 |
| 700 | 0.948853 | 2 3 3 2 2 3 3 2 3 3 3 2 3 2 3 2 2 | 38 | 0.989900 | 2 3 3 2 2 3 3 2 3 3 3 3 2 3 2 2 | 34 |
| | 0.947413 | 1 3 3 2 1 3 3 2 3 3 3 3 3 3 2 2 | 857.968 | 0.989900 | 2 3 3 2 2 3 3 2 3 3 3 2 3 3 2 2 | 824.219 |
| | 0.947413 | 1 3 3 1 2 3 3 2 3 3 3 3 3 3 2 2 | 2265.843 | 0.989899 | 2 3 3 2 2 3 3 3 3 3 3 2 2 3 2 2 | 2321.813 |
| 800 | 0.955288 | 2 3 3 2 2 3 3 3 3 3 3 3 3 2 3 2 3 | 38 | 0.991600 | 2 3 3 2 2 3 3 3 3 3 3 3 3 3 2 3 | 73 |
| | 0.955288 | 2 3 3 2 2 3 3 3 3 3 3 3 3 3 2 3 | 829.485 | 0.991600 | 2 3 3 2 2 3 3 3 3 3 3 3 3 3 2 3 | 1533.266 |
| | 0.954965 | 3 3 3 2 2 3 3 2 3 3 3 3 3 3 3 2 | 2133.500 | 0.991599 | 2 3 3 2 2 3 3 2 3 3 3 3 3 3 3 3 | 2134.782 |

ITER*=no. of generations for optimal solution. CPU**=CPU time (sec) for optimal solution. Total CPU=total CPU time for 100 generations.
 (memory set=5, mutation=0.1, crossover=0.9, population=30, generation=100, affinity=0.1)

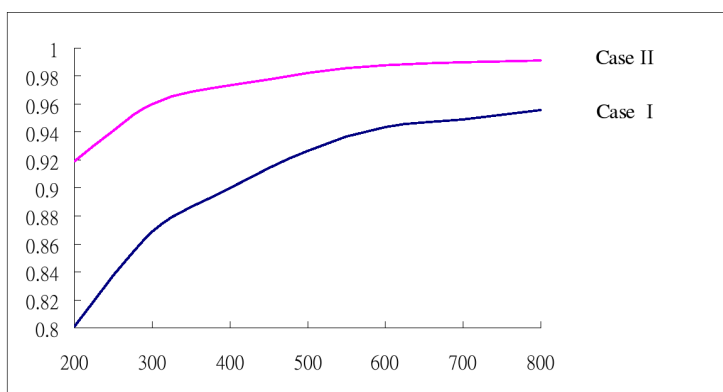


Fig. 5. Optimal reliabilities of traffic-monitor system for test problems of Case I and Case II under various budgets

All results are reported in Table 1 and Fig 5. The programs are coded in C++ and computed by Pentium IV 2.8 GHz PC. From Table 1 and Fig 5, we have:

(1) The CPU time is stable for the immune algorithm. For all cases, it requires less than 2400 sec even the possible combinations is more than 4.29×10^9 for

each test problem. Especially, the immune algorithm converges after about 46 iterations (mean iteration for all test cases).

(2) The presented immune algorithm can obtain multiple optimal solutions for all test problems. For example, for $B=300$ the top-3 system reliabilities are 0.868353, 0.868344 and 0.868097 for Case I.

(3) Fig 5 shows that the optimal solutions tend to use higher levels of monitors with the increase of budgets.

5 Conclusions

In this paper, (i) we have proposed a practical traffic-monitor system which integrates both typical linear $C(k,n:F)$ system and circular $C(k,n:F)$ system, (ii) we have investigated the optimal design of monitors for the traffic-monitor system whose objective function is to minimize the system failure reliability subject to available budget. A new immune algorithm has been proposed to solve such a problem. Numerical results have shown that the presented immune algorithm performs well for all test problems.

References

- [1] Bai, D.S., Yun, W.Y., Cheng, S.W.: Redundancy Optimization of k -out-of- n :G Systems with Common-Cause Failures. IEEE Transactions on Reliability 40 (1991) 56–59
- [2] Chao, M.T., Fu, J.C., Koutras, M.V.: Survey of Reliability Studies of Consecutive- k -out-of- n :F & Related Systems. IEEE Transactions on Reliability 44 (1995) 120–127
- [3] Chao, M.T., Lin, G.D.: Economical Design of Large Consecutive- k -out-of- n :F System. IEEE Transactions on Reliability 33 (1984) 411–413
- [4] De Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer, New York (2002)
- [5] De Castro, L.N., Von Zuben, F.J.: The Clonal Selection Algorithm with Engineering Applications. Workshop Proceedings of the GECCO 2000, (2000) 36–37
- [6] Du, D.Z., Hwang, F.K.: Optimal Assembly of an s -stage k -out-of- n System. SIAM J. Discrete Mathematics 3 (1990) 349–354
- [7] Farmer, J.D., Packard, N.H., Perelson, A.S.: The Immune System, Adaptation, and Machine Learning. Physica 22D (1986) 187–204
- [8] Flynn, J., Chung, C.S.: A Heuristic Algorithm for Determining Replacement Policies in Consecutive k -out-of- n Systems. Computers and Operations Research 31 (2004) 335–348
- [9] Flynn, J, Chung, C.S.: A Bbranch and Bound Algorithm for Computing Optimal Replacement Policies in Consecutive k -out-of- n Systems. Naval Research Logistics 49 (2002) 288–302
- [10] Fu, J.C.: Reliability of Consecutive- k -out-of- n :F Systems with $(k-1)$ Step Markov Dependence. IEEE Transactions on Reliability 35 (1986) 602–606
- [11] Jerne, N.K.: The Immune System. Scientific American (1973) 52–60
- [12] Hsieh, Y.C.: Alternative Approach for Coherent System Reliability with Minimal Cuts/Paths, Working paper (2006)

- [13] Huang, S.J.: Enhancement of Thermal Unit Commitment Using Immune Algorithms Based Optimization Approaches. *Electrical Power and Energy Systems* 21 (1999) 245-252
- [14] Huang, S.J.: An Immune-Based Optimization Method to Capacitor Placement in a Radial Distribution System. *IEEE Transaction on Power Delivery* 15 (2000) 744-749
- [15] Kuo, W., Prasad, R.: An Annotated Overview of System-Reliability Optimization. *IEEE Transactions on Reliability* 49 (2000) 176-187
- [16] Michalewicz, Z. *Genetic Algorithm + Data Structures = Evolution Programs*, Springer-Verlag Berlin Heidelberg, New York (1994)
- [17] Weissman, I.L., Cooper, M.D.: How the Immune System Develops. *Scientific American* 269 (1993) 33-40
- [18] Zuo, M. J., Kuo, W.: Design and Performance Analysis of Consecutive k -out-of- n Structure. *Naval Research Logistics Quarterly* 37 (1990) 203-230

Finding the Optimal Path in 3D Spaces Using EDAs – The Wireless Sensor Networks Scenario

Bo Yuan, Maria Orlowska, and Shazia Sadiq

School of Information Technology and Electrical Engineering,
The University of Queensland, QLD 4072, Australia
{boyuan, maria, shazia}@itee.uq.edu.au

Abstract. In wireless sensor networks where sensors are geographically deployed in 3D spaces, a mobile robot is required to travel to each sensor in order to download the data. The effective communication ranges of sensors are represented by spheres with varying diameters. The task of finding the shortest travelling path in this scenario can be regarded as an instance of a class of problems called Travelling Salesman Problem with Neighbourhoods (TSPN), which is known to be NP-hard. In this paper, we propose a novel approach to this problem using Estimation of Distribution Algorithms (EDAs), which can produce significantly improved results compared to an approximation algorithm.

1 Introduction

In wireless sensor networks where sensors are geographically deployed in 3D spaces (e.g., at different depths below the sea surface), it may not be practical to require sensors to directly coordinate with each other to form a communication network to transfer the data back to the server. A mobile robot is needed in this situation to travel to all sensors to collect the data and come back to its starting location. In order to communicate with each sensor, the robot must be physically within its effective range, which is specified by a sphere. The diameter of the sphere is determined by the current battery level of the sensor and is likely to be different among sensors. The optimization problem in this scenario is to design a path for the robot so that it can collect the data from all sensors while the overall travelling distance is minimized.

In its generic form, this routing problem can be regarded as an instance of a class of problems known as the Travelling Salesman Problem with Neighbourhoods (TSPN) [1] where the neighbourhoods are disjoint spheres. In TSPN, a salesman needs to delivery products to a group of clients living in different places. Instead of waiting at home, each client is willing to meet the salesman within a certain region near their houses. The objective here is to find the shortest trip along which the salesman can meet all clients and come back to the starting location. It is easy to see that TSPN is a generalization of TSP, which is known to be NP-hard [10].

Depending on the properties and the connectivity of the regions, a number of approximation algorithms have been proposed. Arkin and Hassin [1] present the first approximation algorithms for a number of special cases, including parallel unit-segments, translates of a convex region, translates of a connected region and parallel

segments where the ratio between the longest and the shortest segments is bounded by a constant. These algorithms can find a valid tour in polynomial time and its quality is guaranteed to be within a constant factor of the optimal tour. The general idea is to carefully pick up a representative point for each of the regions and then apply a TSP algorithm on this set of points. For the general situations, Mata and Mitchell [9] and Gudmundsson and Levkopoulos [6] present some $O(\log n)$ -approximation algorithms where n is number of regions. Dumitrescu and Mitchell [4] give an $O(1)$ -approximation algorithm for the case of arbitrarily connected regions with comparable diameters and a PTAS (Polynomial Time Approximation Scheme) for the case of disjoint unit disk regions. For problems closely related to our case (i.e., disjoint spheres with possibly varying sizes), de Berg et al. [3] give a constant factor algorithm with approximation factor $12000\alpha^3$ where $\alpha=4$ for disk regions and $\alpha=8$ for 3D spheres. Most recently, Elbassioni et al. [5] show a considerably improved $(9.1\alpha+1)$ -approximation algorithm.

One of the major issues of these approximation algorithms is that, despite their polynomial running time, they are often based on some deterministic procedures and there is an inherent lack of global optimization ability. Also, approximating Euclidean TSPN within $(2-\epsilon)$ has proven to be NP-hard [11]. After all, the performance of these algorithms has only been characterized theoretically in terms of the approximation factors in the worst case, which are often several times worse than the optimal solutions. In real-world situations, simply knowing such a loose bound is obviously of little practical value.

In this paper, we approach this problem from a new perspective by taking advantage of advanced Estimation of Distribution Algorithms (EDAs) [8], which refer to a relatively new paradigm of Evolutionary Algorithms (EAs) [2]. The unique strength of EDAs compared to many other traditional EAs is that they can explicitly capture the dependences among problem variables and use this structural information to conduct more efficient searching, which may often lead to significantly faster convergence speed on challenging optimization problems [8].

2 Methodology

2.1 Problem Specification

The formal definition of the problem is given below. The neighbourhood region corresponding to each sensor is represented by a sphere specified by two parameters: centre e and radius r . A problem instance is then fully specified by the starting position s along with a set of n spheres: $\{s, (e_1, r_1), \dots, (e_i, r_i), \dots, (e_n, r_n)\}$. Note that the radius of each sphere can be significantly different from others, depending on its own power status. Since sensors are supposed to be geographically distributed, here it is assumed that spheres are disjoint from each other. However, as will be shown later, the applicability of the proposed approach is not directly influenced by this factor. Also, we assume that the starting position is not within any sphere as the robot would otherwise have immediate access to the data contained in the sensor. In practice, such sensors may be removed from the problem at the beginning.

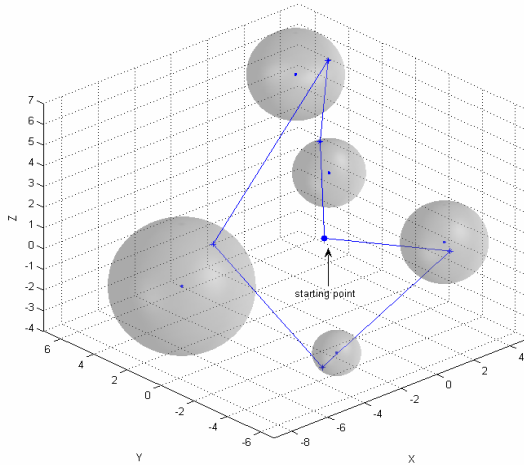


Fig. 1. An example of the layout of the wireless sensors in 3D spaces. There are totally five sensors with each one represented by a sphere of varying sizes. A random path is also shown, which connects all spheres with the starting position.

Figure 1 shows an example of the optimization problem with five sensors as well as a valid tour intersecting with all spheres. Since the robot can start communication with each sensor as long as it reaches the surface of the sphere, the first intersection point between the path of the robot and each sphere is of the major importance. When the data acquisition from a sensor is finished, the direction of movement of the robot will be solely determined by the location of the intersection point on the next sphere. In other words, each TSPN tour is constructed by sequentially connecting the starting position s and n first intersection points on the surfaces of n spherical regions, referred to as hitting points from now on.

Suppose p is the set of n hitting points and π is the permutation over p . A valid tour is then uniquely specified by a tuple $\langle s, p, \pi \rangle$. It is easy to see that there are two groups of parameters to be optimized: the locations of hitting points and their order of appearance (permutation) in the tour. Since the hitting points are distributed on the surface of each sphere, the location of each hitting point can be fully specified by two angles: $\theta \in [0, 2\pi]$ and $\varphi \in [0, \pi]$. The actual coordinate values in the xyz-plane can be calculated by:

$$\begin{aligned}
 x &= x_0 + r \cdot \sin \varphi \cdot \cos \theta \\
 y &= y_0 + r \cdot \sin \varphi \cdot \sin \theta \\
 z &= z_0 + r \cdot \cos \varphi
 \end{aligned}
 \tag{1}$$

In (1), (x_0, y_0, z_0) and r are the centre and radius of the sphere respectively. In this spherical coordinate, φ is the angle between the z-axis and the line connecting the centre (x_0, y_0, z_0) and the point (x, y, z) while θ is the angle between the x-axis and the projection of this line on the xy-plane. Note that there are different ways to define the ranges of θ and φ , which are functionally equivalent.

2.2 Problem Decomposition

According to the problem specification in the last section, for a problem instance with n sensors, there are totally $3n$ parameters to be optimized, including $2n$ continuous parameters specifying the location information and n discrete parameters specifying the permutation. In this section, three different schemes on how EDAs could possibly be applied to this problem will be analysed, taking into account the trade-off between time complexity and global optimization ability.

In the first scheme, all parameters are encoded into the individuals of EDAs, which means that the entire search space is under exploration and the true global optimum is thus guaranteed to be found in theory. The major issue is that the first part of the individual represents a continuous problem while the second part of the individual represents a combinatorial problem. As a result, it would be quite difficult for EAs/EDAs to efficiently handle such individuals and a significant amount of customization may be required to design problem-specific search operators. On the other hand, EAs/EDAs have not been shown to be the most competitive methods as far as TSP is concerned. In other words, they are not particularly good at solving the combinatorial part of the optimization problem. Instead, there exist some dedicated algorithms that are generally more efficient and effective, especially on large TSP instances.

An alternative scheme is to only encode the location information into individuals and EDAs are only responsible for selecting the set of hitting points (i.e., a continuous optimization problem). In this situation, each individual is simply a vector of angles. The quality or fitness of each candidate set of hitting points is measured by the length of the optimal tour based on these points, which is assumed to be given by an external TSP algorithm. Under the assumption that such effective TSP algorithms are available, this scheme can still guarantee finding the global optimum in theory while the dimensionality of the search space to be explored by EDAs can be significantly reduced ($2n$ vs. $3n$). However, since the evaluation of each individual involves solving a TSP instance, this scheme could become very time consuming when there are a relatively large number of sensors.

In order to avoid the above issues, the scheme adopted in our work is based on the following heuristic: find the optimal TSP tour based on the sensor centres and use the same permutation of sensors in the evaluation of each set of hitting points. In other words, we assume that the permutation of spherical regions π^r in the optimal TSPN tour is always in consistence with the permutation of sensor centres π^c in the optimal TSP tour. In practice, this heuristic is certainly not expected to guarantee optimality but to provide a close estimation of π^r . An important question is to determine when this assumption is valid. Although a rigorous analysis is unavailable at this stage, intuitively, if the diameters of spheres are relatively small compared to the distances among spheres, it is very likely that π^r is equal to π^c . After all, when the diameters approach zero, TSPN also gradually approaches TSP.

One of the major advantages of this problem decomposition heuristic is that the dimensionality of the original problem can be reduced from $3n$ to $2n$ and only a single TSP instance needs to be solved. Another advantage is that, given the knowledge of π^r (estimated by π^c), it is now straightforward to precisely evaluate the quality of each set of hitting points and advanced optimization techniques such as EDAs can be conveniently applied. By contrast, although the idea of problem decomposition is also

incorporated in many approximation algorithms, the selection process of candidate hitting points is usually conducted without any clear quality information.

Note that, although the permutation of spheres is determined in advance and fixed in the evaluation of all individuals, the quality of the TSPN tours may still vary significantly with regard to different sets of hitting points. To demonstrate this point, a sampling experiment was conducted with 100,000 random sets of hitting points. The quality (tour length) of each set of hitting points was evaluated based on the problem instance with five sensors (10D search space) shown in Fig. 1. The optimal permutation of sensor centres π^c was found using a brute force search due to the small number of sensors in this problem. The distribution of the quality of the 100,000 samples is shown in Fig. 2 from which it is clear that the quality of TSPN tours was sensitive to the locations of hitting points, even if the permutation was fixed.

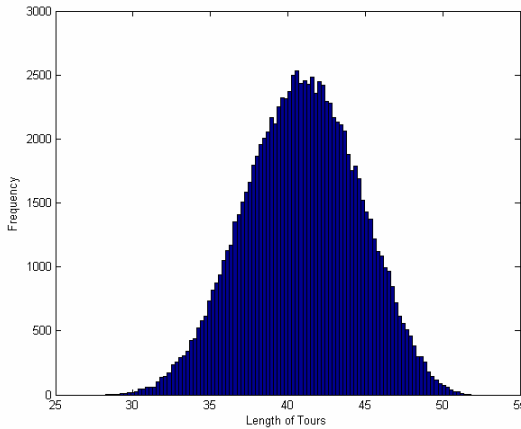


Fig. 2. The distribution of the quality of 100,000 sets of randomly generated hitting points on the five-sensor problem. It shows that there was a large variation on the quality of different sets of hitting points even if the permutation was fixed.

3 Estimation of Distribution Algorithms

The general mechanism of EDAs is to build a statistical model based on a set of promising individuals in each generation. All new individuals are then sampled from the generated model (Table 1). The role of the statistical model is to estimate the structure of the search space where high quality individuals are likely to be found. A continuous EDA named EDA_{mvg} based on a multivariate Gaussian distribution with full covariance structure is used in this paper [8], [12]. In this EDA, the model is specified by the mean vector and the covariance matrix where off-diagonal elements represent the dependence information and all new individuals are generated by sampling from this Gaussian distribution. The advantage of EDA_{mvg} is that it can be implemented very efficiently and still has the capability of capturing complex problem structure. EDAs in this class have shown comparable performance compared to EDAs based on more complex models.

Table 1. The general framework of EDAs

| |
|---|
| Step 1: Initialize the probability model $P_0(X)$, $t=0$ |
| Step 2: Sample a population $O = \{X_1, \dots, X_n\}$ from $P_t(X)$ |
| Step 3: Evaluate individuals in O : $F = \{f(X_1), \dots, f(X_n)\}$ |
| Step 4: Select a subset of the population $O' \subset O$ |
| Step 5: Update the model: $P_t(X) \rightarrow P_{t+1}(X)$ |
| Step 6: $t=t+1$ |
| Step 7: Go to Step 2 until stopping criteria are met |

In (1), using the angles θ and φ , it is possible to specify every point on the surface of each sphere. Since we assume that $\pi^r = \pi^e$, it is possible to reduce the original search space to a smaller area compared to the whole surface. Assume that sphere A is to be visited immediately after sphere B, the distribution of all possible hitting points on sphere A can be determined by the diameters of A and B as well as their relative spatial location (see Fig. 3 for a 2D example). In other words, there is always a “dark side” on each sphere where no hitting points could possibly be located. It is easy to image that, if A and B are of the same size, the dark side on A will account for at least 50% of the surface of A (it also depends on the distribution of hitting points on B).

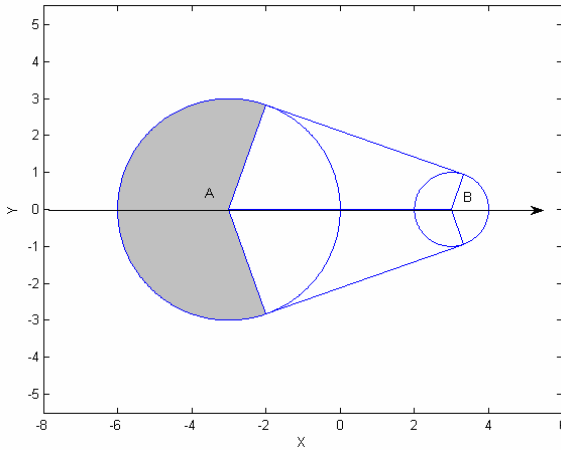


Fig. 3. A 2D example of the “dark side” on sphere A visited immediately after sphere B

However, it is not convenient to specify the feasible search space analytically in terms of θ and φ and make sure all individuals are generated within this area. Alternatively, each individual is checked before evaluation to make sure that the hitting points are valid. Suppose that the current hitting point on sphere B is P_B and the candidate hitting point on sphere A is P_A . By connecting P_A and P_B with a line L , the intersection point(s) between L and A can be worked out. If there exists another intersection point P'_A that is closer to P_B , it will replace P_A as the new hitting point on A as P_A is not the first intersection point (hitting point) on sphere A.

The intersection points can be found by solving for u the following two sets of equations:

$$\begin{aligned} \text{Line } L: \quad & x = P_{B,x} + u \cdot (P_{A,x} - P_{B,x}) \\ & y = P_{B,y} + u \cdot (P_{A,y} - P_{B,y}) \\ & z = P_{B,z} + u \cdot (P_{A,z} - P_{B,z}) \end{aligned} \tag{2}$$

$$\text{Sphere } A: (x - x_A)^2 + (y - y_A)^2 + (z - z_A)^2 = r_A^2. \tag{3}$$

4 Experiments

In this section, we empirically evaluated the performance of the proposed techniques for TSPN problems compared to an approximation algorithm called Algorithm A by Elbassioni et al. [5]. The major motivation was to provide some preliminary results to justify the soundness of our methods. More comprehensive experimental studies will be conducted as the future work.

The basic procedure of Algorithm A is to progressively select a hitting point for each sphere and then construct a TSP tour based on the set of selected hitting points. More specifically, all spheres are initially sorted in ascending order based on their diameters. The hitting point on the smallest sphere is selected randomly. In our case, the starting position s is regarded as a zero-diameter sphere and will always be selected as the first hitting point. The hitting point on each subsequent sphere is selected as the point with the minimum distance to the current set of hitting points.

Note that both TSPN algorithms require an external TSP algorithm to either find the optimal permutation of sensor centres (our method) or construct the final tour based on a set of hitting points (Algorithm A). The Lin-Kernighan heuristic [7] was used in this paper, while any other TSP algorithms capable of handling 3D Euclidean spaces will also be appropriate.

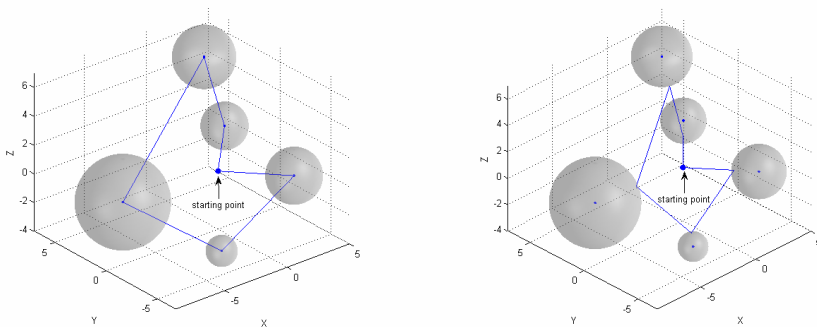


Fig. 4. The optimal TSP tour (length=39.0140) of the five-sensor problem found by a brute force search (*left*) and the TSPN tour (length=27.2908) constructed by Algorithm A (*right*)

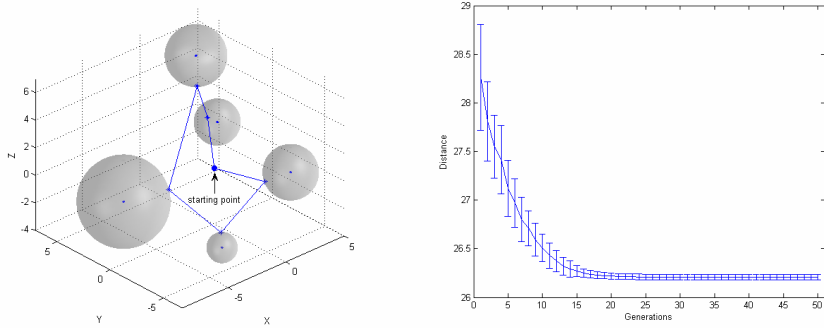


Fig. 5. A typical TSPN tour (length = 26.1830) of the five-sensor problem found by the EDA method (*left*) and the experimental results averaged over 25 independent trials (*right*)

The first experiment was based on the five-sensor test problem shown in Fig. 1. Due to its simplicity, a brute force search was used to solve the TSP components in both algorithms. Figure 4 (left) shows the optimal TSP tour based on sensor centres with length 39.0140. Since the TSP tour is also a valid TSPN tour, it can be used as a benchmark against TSPN algorithms. The corresponding TSPN tour with length 27.2908 constructed by Algorithm A (i.e., an optimal TSP tour found through the brute force search based on the set of hitting points selected by Algorithm A) is shown in Fig. 4 (right).

Next, based on the permutation of spheres in the optimal TSP tour, the EDA was applied to search for the set of hitting points. The parameter values were chosen as population size=100, number of generations=50, truncation selection with selection pressure=0.3 (select the top 30% individuals). No specific parameter tuning was conducted and the above values were largely based on recommended values and a few preliminary trials.

Figure 5 (left) shows a typical TSPN tour constructed by the EDA approach with length 26.1830. In order to demonstrate the robustness of our method, 25 independent trials were conducted and the mean performance is plotted in Fig. 5 (right), together with error bars showing one standard deviation above and below the mean value. It is easy to see that our approach could reliably find solutions better than the approximation algorithm. Note that there is no parameter to be tuned in Algorithm A and its performance is deterministic.

To further verify the performance of the EDA method, a much more challenging test problem with 25 sensors was randomly generated, as shown in Fig. 6. The starting point was at the origin while sensors were randomly distributed within $[-20, 20]$ in each dimension. The radius of each sphere was randomly chosen within $[1, 4]$ and special care has been taken to make sure that spheres were disjoint from each other. The optimal TSP tour of this problem had length 304 with rounding (the TSP algorithm in use assumes that distances among nodes are integral) while the TSPN tour constructed by Algorithm A had length 271 (with rounding). By contrast, with a larger population size (1000) and more generations (100), the EDA method could again reliably find solutions of significantly improved quality. A typical TSPN tour is shown in Fig. 6 with length 248 (with rounding).

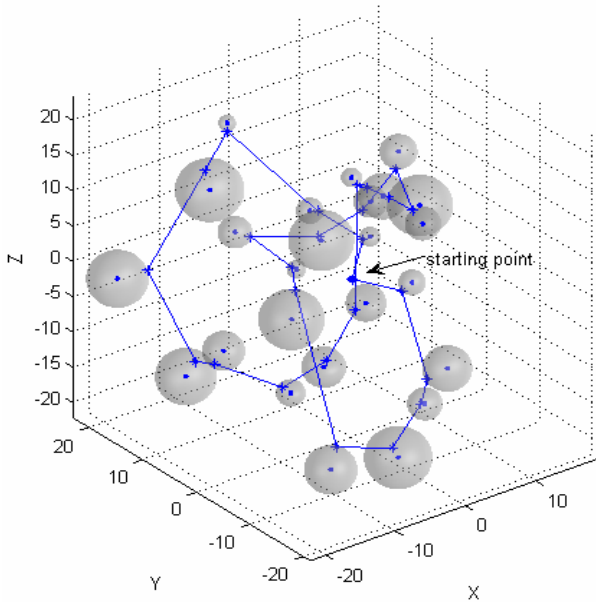


Fig. 6. A random test problem with 25 sensors and a typical TSPN tour (length = 248) found by the EDA method

5 Conclusions

In this paper, we presented some preliminary work on a novel technique for finding the optimal path in 3D spaces, which was motivated by the data acquisition problem in wireless sensor networks. The core idea is to decompose the original problem into a TSP problem and a continuous optimization problem. In the TSP component, an optimal tour is constructed by an external TSP algorithm based on the sensor centres. The permutation of sensors is then used as the estimation of the permutation of spheres in the TSPN tour. Consequently, the EDA is dedicated to searching for the optimal set of hitting points, represented by a vector of angles.

In the numerical simulations of two test problems, the performance our method has shown to be robust and encouraging and it could significantly improve the quality of the tours constructed by the latest TSPN algorithm.

As to the future work, a major step is to establish some theoretical analysis on the effectiveness of the heuristic (estimation of π^f by π^c) used in our method. It would be favourable to be able to formally investigate the analytical properties of this heuristic in order to understand and quantify its influence on the quality of tours.

Although the EDA performed reasonably well in the experiments, an important task is to further characterize the structure of this type of problems to have a better understanding on what kind of optimization algorithms are mostly applicable and how to incorporate more problem-specific knowledge into these algorithms in order to achieve better performance.

References

1. Arkin, E. M., Hassin, R.: Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Mathematics*, Vol. 55 (3), (1994) 197-218.
2. Bäck, T., Fogel, D. B., Michalewicz, Z., Eds.: *Handbook of Evolutionary Computation*, IOP Publishing Ltd and Oxford University Press (1997).
3. de Berg, M., Gudmundsson, J., Katz, M. J., Levcopoulos, C., Overmars, M. H., van der Stappen, A. F.: TSP with Neighborhoods of Varying Size. *Journal of Algorithms*, Vol. 57 (1), (2005) 22 - 36.
4. Dumitrescu, A., Mitchell, J. S. B.: Approximation algorithms for TSP with neighborhoods in the plane. *Journal of Algorithms*, Vol. 48 (1), (2003) 135 - 159.
5. Elbassioni, K., Fishkin, A., Mustafa, N., Sitters, R.: Approximation Algorithms for Euclidean Group TSP. In the 32nd International Colloquium on Automata, Languages and Programming, (2005) 1115-1126.
6. Gudmundsson, J., Levcopoulos, C.: A fast approximation algorithm for TSP with neighborhoods. *Nordic Journal of Computing*, Vol. 6 (4), (1999) 469 - 488.
7. Helsgaun, K.: An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic. *European Journal of Operational Research*, Vol. 126 (1), (2000) 106-130.
8. Larrañaga, P., Lozano, J. A., Eds.: *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, Kluwer Academic Publishers (2001).
9. Mata, C. S., Mitchell, J. S. B.: Approximation algorithms for geometric tour and network design problems (extended abstract). In the eleventh annual symposium on computational geometry, (1995) 360 - 369.
10. Papadimitriou, C. H.: The Euclidean travelling salesman problem is NP-complete. *Theoretical Computer Science*, Vol. 4 (3), (1977) 237-244.
11. Safra, S., Schwartz, O.: On the complexity of approximating tsp with neighborhoods and related problems. *Computational Complexity*, Vol. 14 (4), (2006) 281 - 307.
12. Yuan, B., Gallagher, M.: On the Importance of Diversity Maintenance in Estimation of Distribution Algorithms. In the 2005 Genetic and Evolutionary Computation Conference, (2005) 719-726.

Evidential Reasoning Based on Multisensor Data Fusion for Target Identification

Xin Wang^{1,2}, Yunxiao Wang^{1,2}, Xiao Yu³, Zhengxuan Wang^{1,2},
and Yunjie Pang¹

¹ College of Computer Science and Technology
Jilin University, 130012, Changchun, China
w_x@jlu.edu.cn

² Key Laboratory of Symbolic Computation and Knowledge Engineer of Ministry of
Education, Jilin University, 130012, Changchun, China

³ Aviation University of Air Force, 130022, Changchun, China

Abstract. Air target identification is an important issue in threat warning, airline security and surveillance. To obtain accuracy and reliability, the multisensor is used to give multiple sources information. Thus, an algorithm to fuse the information from the multisensor is needed. The (Dempster-Shafer) evidence theory is a generalization of Bayesian statistics. Evidential reasoning is suited to a range of decision-making activities. But it is invalid when dealing with conflicting probabilities. In this paper, a new weighted D-S combination rule is proposed to solve the conflicting management in the air target identification system. In the weighted method presented here, it is to modify evidences rather than to modify the combination rule. The rationality and effectiveness of the weighted method are evaluated by the target identification system.

1 Introduction

Air target identification provides the basis for Battlefield Situation/Threat Assessment. Due to the influence of sensor precisions, components of the integrated system and the external circumstance, the identification by a single sensor is uncertain and not suitable to air target identification. Gathering the multi-source data with multisensor to acquire the complete information of the situation and characteristics of the observing target, and producing meaningful new fused information can improve the precision of target identification and categorization.

Evidence theory is a generalization of Bayesian statistics [1,2]. It involves defining a belief function over a hypothesis set. Belief function satisfies the weaker axiom that is not appear in probability theory. It can distinguish uncertainty and ignorance. Evidence theory provides not only flexible representation of evidence, but also simple and intuitive reasoning rules to update belief function. The consideration of ignorance on belief assignment is similar to the thinking habit of human expert. It has been widely used in information fusion and expert systems [3,4].

In the air target identification system, the category probabilities that acquired by different sensors for the same target might be conflict with each other as the

fault of the sensor or as the human factor. If applying the evidence combination rule of D-S evidence theory to update the belief function, the highly conflicting probabilities tend to produce counterintuitive results [6].

In this paper, we propose a new method for handling cases when the beliefs are conflicting. Unlike the previous solve strategies [8,9], this method analyzes the input evidence without trying to modify the combination rule. The conflicts indicate that the evidence offered by different sensors should not be trusted for the same extent. The new approach preprocesses the input evidence: if one or minority of evidence conflict with the most of evidence, the method reduces the credibility of the minority of evidence, and increases their ignorance degree so that it could weaken the influence on fusion result.

As illustrated by the simulated data, the method is feasible to multisensor information fusion for the air target identification, and it can improve the identification precision and increase the reliability of target classification.

2 D-S Evidence Theory

Evidence theory is a generalization of the venerable Bayesian statistics. In evidence theory, evidence is described in terms of evidential functions. The commonly used functions include: basic probability assignment function, belief function and plausibility function.

2.1 Evidential Functions

Definition 1. Let Θ be the frame of discernment. 2^Θ be the set of all subsets that represent propositions in Θ . A function $m : 2^\Theta \rightarrow [0, 1]$ is called a mass function if it satisfies

$$m(\emptyset) = 0, \sum_{X \subseteq \Theta} m(X) = 1. \tag{1}$$

A mass function is a basic probability assignment (BPA) function to all subsets X of Θ . The quantity $m(A)$ is called A 's mass or basic probability value. It represents the exact amount of belief committed to the proposition represented by subset A of Θ . A focus of m is a subset $A \subseteq \Theta$ on which m is positive, i.e., $m(A) \neq 0$.

Definition 2. A function $bel : 2^\Theta \rightarrow [0, 1]$ is a belief function if it satisfies

$$bel(A) = \sum_{X \subseteq \Theta, X \subseteq A} m(X). \tag{2}$$

The quantity $bel(A)$ can be interpreted as a measure of one's belief that hypothesis A is true. Where $bel(\emptyset)=0, bel(\Theta)=1$.

Definition 3. A function $pls : 2^\Theta \rightarrow [0, 1]$ is a plausibility function if it satisfies

$$pls(A) = \sum_{X \subseteq \Theta, X \cap A \neq \emptyset} m(X). \tag{3}$$

It is easy to say that $pls(\emptyset)=0, pls(\Theta)=1$.

Also, we have $bel(A) + bel(\bar{A}) \leq 1$, or $bel(A) \leq pls(A)$ for all $A \subseteq \Theta$. For a given subset A , the information contained in the evidential functions and may be conveniently represented by the belief interval $[bel(A), pls(A)]$. A plausibility function is also called an upper probability function, while a belief function is called a lower probability function. Here $bel(A)$ gives the degree to which the current evidence supports A , and $pls(A) = 1 - bel(\bar{A})$ gives the degree to which A remains plausible. We call $bel(A)$ the lower probability of A , $pls(A)$ the upper probability of A . The difference $pls(A) - bel(A)$ represents the residual ignorance:

$$ignorance(A) = pls(A) - bel(A). \tag{4}$$

2.2 D-S’s Rule of Combination

The fundamental operation in evidential reasoning is the orthogonal sum of evidential functions. It is known as D-S’s rule for combining evidence. Let m_1, m_2, \dots, m_n be n BPA functions on the same frame Θ , the orthogonal sum $m = m_1 \oplus m_2 \oplus \dots \oplus m_n$ is defined as

$$m(A) = (1 - k)^{-1} \sum_{\cap A_i = A} \prod_{1 \leq i \leq n} m_i(A_i). \tag{5}$$

where $k = \sum_{\cap A_i = \emptyset} \prod_{1 \leq i \leq n} m_i(A_i)$ is the conflict factor; and $1/(1 - k)$ is called the normalization constant, it normalizes the new distribution by reassigning any probability mass which is assigned to the empty set by the combination.

2.3 Evidence Conflict Management

The Eq.(5) is invalidate while the denominator $1 - k = 0$, it implies complete contradiction between m_1, m_2, \dots, m_n . If the pieces of evidence are highly conflicting, the combination rule will assign the 100% belief to a minor probability hypothesis and produce a counterintuitive result. A typical example given by Zadeh [6] is as follows:

Example 1. Let $\Theta = \{A, B, C\}$, assume m_1, m_2 be two mass functions on the frame of discernment Θ : $m_1(A) = 0.9, m_1(B) = 0.1, m_2(B) = 0.1, m_2(C) = 0.9$.

These two pieces of evidence m_1 and m_2 are largely conflicting. By using D-S’s rule, we have

$$m(A) = 0, m(B) = 1, m(C) = 0.$$

This means the combined result agrees with B no matter that it is only weakly supported by the respective original pieces.

Various alternative methods have been proposed to solve the problem of conflict management. Yager [8] presented an alternative rule of combination (denoted by \perp) as follows:

Definition 4. Assume m_1, m_2 be two mass functions on the frame of discernment. Let $\{A_i\}, \{B_j\}$ be their sets of focal elements. Then

$$(m_1 \perp m_2)(\emptyset) = 0$$

$$(m_1 \perp m_2)(X) = \sum_{A_i \cap B_j = X} m_1(A_i)m_2(B_j), \text{ for } \emptyset \subset X \subset \Theta$$

$$(m_1 \perp m_2)(\Theta) = \sum_{A_i \cap B_j = \Theta} m_1(A_i)m_2(B_j) + E. \tag{6}$$

where $E = \sum_{A_i \cap B_j = \emptyset} m_1(A_i)m_2(B_j)$ (the total conflict).

With the use of this alternative rule, the total conflict E is put back into the set Θ . Thus, by using Yager’s rule to *Example 1*, we have

$$m(A) = 0, m(B) = 0.01, m(C) = 0, m(\Theta) = 0.99.$$

3 The Weighted D-S Combination Method

In air target identification system, the conflict of evidence is usually caused of the invalidation sensor with many factors such as weather, enemy jamming or sensor fault. It means that the sensors should not be trusted equally. Thus, we proposed a new combination method to fuse the conflict evidence. In this method, we haven’t tried to find out an alternative combination rule and acquire a more reasonable result. But, we firstly adjust each sensor’s BPAs before they are submitted to the D-S reasoning system. The brief steps of the new conflict management are described as following: first the distance from the BPAs to the average is used to decide the weight of each sensor; then justify each sensor’s BPAs by the credibility weight; finally the BPAs adjusted are combined using D-S combination rule.

Suppose there are u sensors, v categories of target. Then for an observed target, we will acquired $u \times v$ BPAs.

$$\begin{bmatrix} m_{11} & m_{12} & \dots & m_{1v} \\ m_{21} & m_{22} & \dots & m_{2v} \\ \dots & \dots & \dots & \dots \\ m_{u1} & m_{u2} & \dots & m_{uv} \end{bmatrix}$$

The BPAs adjustment procedure we proposed is shown following:

- (1) Calculate the average masses of the evidence:

$$M_{MED}^j = \frac{1}{u} \sum_{i=1}^u m_{ij} \quad (j = 1, 2, \dots, v) \tag{7}$$

(2) Calculate the distance between the BPAs of each sensor and the average masses:

$$D_i = \sum_{j=1}^v |m_{ij} - M_{MED}^j| \quad (i = 1, 2, \dots, u) \tag{8}$$

(3) Compute the credibility weight of each sensor:

$$w_i = \frac{W_i}{W_{max}} \tag{9}$$

where $W_i = 1/D_i$. It means the closer a piece of evidence to the average masses, the higher reliability of this evidence. If a piece of evidence far from the average masses, it means the sensor doesn't work-well, then we assign a lower reliability to the evidence. Where $W_{max} = max(W_i)$ ensures the credibility weight w_i is in the range of $[0, 1]$.

(4) Adjust each sensor's BPAs before they are submitted to the D-S reasoning system. The weight adjustment process is expressed as:

$$\begin{aligned} m_i^*(A) &= w_i m_i(A), \text{ for } A \subset \Theta \\ m_i^*(\Theta) &= w_i m_i(\Theta) + 1 - w_i \end{aligned} \tag{10}$$

where the weighting factor w_i is in the range of 0.0 and 1.0, the $m_i(\Theta)$ stands for the probability assigned to the acknowledgement of ignorance, and the term $m_i^*(A)$ indicates the adjusted BPAs to be submitted to the D-S reasoning system. As the weighting factor is always equal to or less than 1.0, this differential credibility effectively lowers the BPAs assigned to the non-zero hypothesis by each sensor and correspondingly raises the value assigned to the ignorance set. This will mitigate conflicts among groups of evidence.

4 Air Target Identification

In an air battle system [10], there may be 5 kinds of target: battle aircrafts, offensive aircrafts, bombers, EW or AWACS and other aircrafts (e.g. helicopter, missile). They are represented by a, b, c, d, e . Also, three kinds of sensor ESM, EO, IR are used to collect information RF, PW, IR respectively. The optical system is used to observe the visible light information, which represented as OB .

4.1 Probability Assignment Functions

Now, we have frame of discernment $\Theta = \{a, b, c, d, e\}$. According to RF, PW, IR, OB , we have four mass functions $m_{RF}, m_{PW}, m_{IR}, m_{OB}$ as follows:

| X | $\{a\}$ | $\{b\}$ | $\{c\}$ | $\{d\}$ | $\{e\}$ |
|----------|---------|---------|---------|---------|---------|
| m_{RF} | 0.50 | 0.20 | 0.15 | 0.15 | 0.00 |
| m_{PW} | 0.60 | 0.15 | 0.25 | 0.00 | 0.00 |
| m_{IR} | 0.00 | 0.00 | 0.20 | 0.40 | 0.40 |
| m_{OB} | 0.60 | 0.30 | 0.10 | 0.00 | 0.00 |

Analyzing the source data, we can find that the piece of evidence m_{IR} contradict to the other three pieces of evidence. Following the *majority rule*, it will be a good choice to decrease the influence of this evidence to the combination result by adjusting the evidence.

4.2 Justify the BPA Function

Follow Eq.(7) acquired the average masses: $m_{MED}\{a\}=0.425$, $m_{MED}\{b\}=0.1625$, $m_{MED}\{c\}=0.175$, $m_{MED}\{d\}=0.1375$, $m_{MED}\{e\}=0.1$. And obtain the credibility weight by using Eq.(8)-(9),

$$w_{RF} = 1, w_{PW} = 0.50, w_{IR} = 0.21, w_{OB} = 0.40 .$$

Then reassign the BPAs of each sensor by using Eq.(10):

| X | $\{a\}$ | $\{b\}$ | $\{c\}$ | $\{d\}$ | $\{e\}$ | Θ |
|------------|---------|---------|---------|---------|---------|----------|
| m_{RF}^* | 0.50 | 0.20 | 0.15 | 0.15 | 0.00 | 0.00 |
| m_{PW}^* | 0.30 | 0.07 | 0.13 | 0.00 | 0.00 | 0.50 |
| m_{IR}^* | 0.00 | 0.00 | 0.05 | 0.08 | 0.08 | 0.79 |
| m_{OB}^* | 0.24 | 0.12 | 0.04 | 0.00 | 0.00 | 0.60 |

4.3 Combine $m_{RF\&PW\&IR\&OB} = m_{RF}^* \oplus m_{PW}^* \oplus m_{IR}^* \oplus m_{OB}^*$

Substitute the processed BPAs into Eq.(5), we have:

$$\frac{X}{m_{RF\&PW\&IR\&OB}(X)} \mid \begin{matrix} \{a\} & \{b\} & \{c\} & \{d\} & \{e\} & \Theta & \sum \\ 0.62 & 0.15 & 0.14 & 0.09 & 0.00 & 0.00 & 1 \end{matrix}$$

4.4 Comparison of Combination Results

The combination results of several representative combination methods for the air target identification system are listed in Table 1. To classical D-S combination rule, because the BPA of category $\{a\}$, $m_{IR}(\{a\})$ is zero, the combination BPA is always zero. Although all the BPAs to $\{c\}$ are very small, the combination result is the only category $\{c\}$, because none of the evidence is completely negated. Yager’s method assigned all conflicts to ignorance. So the combination BPAs to every category are all small while the BPAs assigned to ignorance set Θ is very large. From the combination result, one can’t make the category decision.

The proposed method justifies each sensor’s BPAs by the weight achieved from the distance to average. The scheme effectively lowers the basic probability assigned to the unreliable evidence and correspondingly raises the value assigned to the ignorance set thereby lowers the influence of the evidence to the combination result. From the combination result acquired by the weighted D-S combination rule, the BPA of $m_{IR}(\{a\})$ is zero, the $m_{RF\&PW\&IR\&OB}(\{a\})$ is 0.63. The result reflects the tendency of most evidence. We can conclude that the target belongs to category $\{a\}$ with the most probability.

Table 1. Comparison of different combination results

| | <i>RF&PW</i> | <i>RF&PW&IR</i> | <i>RF&PW&IR&OB</i> |
|-----------------------------------|------------------------------|---------------------------------|------------------------------------|
| D-S combination rule | $m_{RF&PW}\{a\} = 0.82$ | $m_{RF&PW&IR}\{a\} = 0$ | $m_{RF&PW&IR&OB}\{a\} = 0$ |
| | $m_{RF&PW}\{b\} = 0.08$ | $m_{RF&PW&IR}\{b\} = 0$ | $m_{RF&PW&IR&OB}\{b\} = 0$ |
| | $m_{RF&PW}\{c\} = 0.10$ | $m_{RF&PW&IR}\{c\} = 1$ | $m_{RF&PW&IR&OB}\{c\} = 1$ |
| | $m_{RF&PW}\{d\} = 0$ | $m_{RF&PW&IR}\{d\} = 0$ | $m_{RF&PW&IR&OB}\{d\} = 0$ |
| | $m_{RF&PW}\{e\} = 0$ | $m_{RF&PW&IR}\{e\} = 0$ | $m_{RF&PW&IR&OB}\{e\} = 0$ |
| Yager's combination rule | $m_{RF&PW}\{a\} = 0.30$ | $m_{RF&PW&IR}\{a\} = 0$ | $m_{RF&PW&IR&OB}\{a\} = 0.22$ |
| | $m_{RF&PW}\{b\} = 0.03$ | $m_{RF&PW&IR}\{b\} = 0$ | $m_{RF&PW&IR&OB}\{b\} = 0.11$ |
| | $m_{RF&PW}\{c\} = 0.04$ | $m_{RF&PW&IR}\{c\} = 0.13$ | $m_{RF&PW&IR&OB}\{c\} = 0.05$ |
| | $m_{RF&PW}\{d\} = 0$ | $m_{RF&PW&IR}\{d\} = 0.25$ | $m_{RF&PW&IR&OB}\{d\} = 0$ |
| | $m_{RF&PW}\{e\} = 0$ | $m_{RF&PW&IR}\{e\} = 0.25$ | $m_{RF&PW&IR&OB}\{e\} = 0$ |
| | $m_{RF&PW}\{\Theta\} = 0.63$ | $m_{RF&PW&IR}\{\Theta\} = 0.37$ | $m_{RF&PW&IR&OB}\{\Theta\} = 0.62$ |
| New weighted D-S combination rule | $m_{RF&PW}\{a\} = 0.58$ | $m_{RF&PW&IR}\{a\} = 0.57$ | $m_{RF&PW&IR&OB}\{a\} = 0.63$ |
| | $m_{RF&PW}\{b\} = 0.17$ | $m_{RF&PW&IR}\{b\} = 0.17$ | $m_{RF&PW&IR&OB}\{b\} = 0.16$ |
| | $m_{RF&PW}\{c\} = 0.14$ | $m_{RF&PW&IR}\{c\} = 0.14$ | $m_{RF&PW&IR&OB}\{c\} = 0.12$ |
| | $m_{RF&PW}\{d\} = 0.11$ | $m_{RF&PW&IR}\{d\} = 0.12$ | $m_{RF&PW&IR&OB}\{d\} = 0.09$ |
| | $m_{RF&PW}\{e\} = 0$ | $m_{RF&PW&IR}\{e\} = 0$ | $m_{RF&PW&IR&OB}\{e\} = 0$ |

5 Conclusion

A new strategy is proposed to solve the invalidation problem of air target identification. As the example illustrated, the modified method is an efficient solution for conflicting management. Compared to other combination rules, the proposed method is more similar to the human expert. The precision and reliability of the system is improved.

Acknowledgement. This work was supported by the Specialized Research Fund for the Doctoral Program of Higher Education (No. 20060183041).

References

1. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press, Princeton, New Jersey. (1976)
2. Guan, J. W., Bell, D. A.: Evidence theory and its applications, Vol.1, 2. Studies in Computer Science and Artificial Intelligence 7, 8, Elsevier, The Netherlands. (1991, 1992)
3. Walz, E., Llinas, J.: Multisensor data fusion. Boston: Artech House. (1990)
4. Hall, D. L., Llinas, J.: An introduction to multisensor data fusion. Proceedings of IEEE **85** (1997) 6-23
5. Hall, D. L., McMullen, S. A. H.: Mathematical techniques in multisensor data fusion. Boston: Artech House. (2004)
6. Zadeh, L. A.: Review of Shafer's a mathematical theory of evidence. AI Magazine. **5** (1984) 81-83

7. Zadeh, L. A.: A simple view of the Dempster-Shafer theory of evidence and its implication for the rule of combination. *AI Magazine*. **85** (1986) 85–90
8. Yager, R. R.: On the Dempster-Shafer framework and new combination rules, *Information Sciences*. **41** (1987) 93–137
9. Hau, H. Y., Kashyap, R. L.: Belief combination and propagation in a lattice-structured inference network, *IEEE Trans. On Systems, Man, and Cybernetics*. **20** (1990) 45–57
10. Liu, T. M.; Xia, Z. X.; Xie, H. C.: *Data fusion and its applications*. National Defense Press, Beijing, China (1998)

A Simple and Compact Algorithm for the RMQ and Its Application to the Longest Common Repeat Problem*

Inbok Lee¹ and Ha Yoon Song²

¹ School of Electronic, Telecommunication, and Computer Engineering
Hankuk Aviation University, Republic of Korea

inboklee@hau.ac.kr

² Department of Computer Engineering
Hongik University, Republic of Korea

hayoon@wow.hongik.ac.kr

Abstract. The Range Minimum Query (RMQ) problem is to find the smallest element in an array for given range (a, b) . We propose a simple and compact algorithm for this problem when the queries are sorted in ascending order. Then we show how to use this algorithm for the generalised longest common repeat problem [14]. Our algorithm is easy to understand and implement and requires much smaller memory.

1 Introduction

The Range Minimum Query (RMQ) problem is defined as follows.

Definition 1. *Given an array $A[1..n]$ of integers and a range (a, b) ($1 \leq a \leq b \leq n$), report the position k ($a \leq k \leq b$) where $A[k]$ is the smallest in $A[a..b]$.*

Note that we may be interested in not the position k but the value $A[k]$.

The RMQ problem has a lot of applications in string processing and computational biology. Related problems include the Lowest Common Ancestor (LCA) problem. In this problem, given two leaves in a tree, we want to find the *lowest* node whose subtree includes them. A naïve algorithm for the problem is to scan the array $A[a..b]$ from left to right and report the position where the smallest element is located. It runs in $O(n)$ time in the worst case. But in many cases we are not just asking one query. Therefore we need an efficient algorithm for the RMQ.

Using precomputation we can obtain a better algorithm. There are a few works on these problems [3, 7, 15, 11, 6, 2]. We briefly explain a simple method in [2]. We use an array $M[n][\log n]$. For each element $A[a]$ ($1 \leq a \leq n$), we find the minimum element in $A[a..a + 2^k - 1]$ ($1 \leq k \leq \log n$) and store it in $M[a][k]$. It takes $O(n \log n)$ time and space. For answering the RMQ query for $A[a..b]$, we divide the range into two: $(a, a + 2^{\log(b-a)} - 1)$ and $(b - 2^{\log(b-a)} + 1, b)$.

* This work was supported in part by the Seoul R&BD Program funded by the Seoul Development Institute (DRI).

Then we find these values from the array M and report the smaller one. $O(n)$ -time preprocessing and $O(1)$ -time query with $O(n)$ space is the best complexity [16]. Although these algorithms have good time/space complexity and are elegant, they are not easy to understand. Furthermore, all these algorithms are based on the assumption that all the entries of A is known in advance. In some applications it doesn't hold. The array A can be computed on the fly and we maintain only a sliding window over A . In that case, range minimum queries are restricted in the sliding window. We also assume that the window moves from left to right: left ends of ranges for RMQ are in ascending order.

Repetitions play an important role in real world applications such as Bioinformatics, computer-aided music analysis, cryptanalysis, and so on. Especially we focus on analyzing genomic sequences in Bioinformatics. It is assumed that repetitions in genomic sequences are related to genetic diseases. Also, they can give us clues to RNA secondary structures. They may be used in multiple alignment. These problems gets tricky when we consider a set of genomic sequences. Some repeats appear frequently (i.e. more than twice) in the sequence. Therefore it would be better if we can specify the number of times a repeat can appear in special sequences. And, some repeats may not be common to all the sequences in the set: they may appear only in some subset of the sequences.

In [13], an algorithm based on the generalised suffix tree was proposed. It was the first approach, but it was not easy to implement. And it requires too much memory. In [14], another algorithm based on the suffix array was proposed. The definition of problem was modified to handle the general case (the problem in [13] cannot set the number of times that a repeat can appear in a string). It is much easier to implement and used less memory.

2 Preliminaries

Since our main motivation is Bioinformatics, we assume that the alphabet $\Sigma = \mathbf{a, c, g, t}$ and a bar represents its Watson-Crick pair: $\bar{\mathbf{g}} = \mathbf{c}, \bar{\mathbf{c}} = \mathbf{g}, \bar{\mathbf{a}} = \mathbf{t},$ and $\bar{\mathbf{t}} = \mathbf{a}$

Let T be a string over Σ . $T[i]$ denotes i -th character of T and $T[i..j]$ is the substring $T[i]T[i + 1] \cdots T[j]$ of T . We denote the *reverse* of T by \overleftarrow{T} and the *reverse complemented* of T by $\overleftarrow{\bar{T}}$. To get $\overleftarrow{\bar{T}}$, we first make \overleftarrow{T} and replace each character with its Watson-Crick pair.

A *repeat* of T is a substring of T which appears at least twice in T . There are three kinds of repeats.

- DIRECT REPEAT: A string p is called a *direct repeat* of T if $p = T[i..i + |p| - 1]$ and $p = T[i'..i' + |p| - 1]$, for some $i \neq i'$.
- INVERTED REPEAT: A string p is called an *inverted repeat* of T if $p = T[i..i + |p| - 1]$ and $\bar{p} = T[i'..i' + |p| - 1]$, for some $i \neq i'$.
- MIRROR REPEAT: A string p is called a *mirror repeat* (or *reverse repeat*) of T if $p = T[i..i + |p| - 1]$ and $\overleftarrow{\bar{p}} = T[i'..i' + |p| - 1]$, for some $i \neq i'$.

Once we define repeats, our problem can be defined as followed.

Problem 1. [14] Given a set of strings $\mathcal{U} = \{T_1, T_2, \dots, T_\ell\}$, a set of positive integers $\mathcal{D} = \{d_1, d_2, \dots, d_\ell\}$, and a positive integer k , the GENERALISED LONGEST COMMON REPEAT PROBLEM is to find the longest string w which satisfies two conditions: (a) There is a subset \mathcal{U}' of \mathcal{U} such that w appears at least d_i times in every string T_i in \mathcal{U}' , and (b) $|\mathcal{U}'| = k$.

Note that we can restrict the number of times that a repeat can appear in each string. Even we can let the “repeat” appear just once in some strings.

The *suffix array* of a text T is a well-known indexed structure for the string. Basically it is the sorted array $s[1..|T|]$ of all the suffixes of T . It means that $s[k] = i$ if and only if $T[i..|T|]$ is the k -th suffix of T . We also define the auxiliary *LCP array* as an array of the length of the longest common prefix between each substring in the suffix array and its predecessor, and define $\text{lcp}(a, b) = \min_{a \leq i \leq b} \text{lcp}[i]$ with the following properties.

Fact 1. $\text{lcp}(a, b) \leq \text{lcp}(a', b')$ if $a \leq a'$ and $b \geq b'$.

Fact 2. The length of the longest common prefix of $T[s[a]..|T|]$, $T[s[a + 1]..|T|]$, \dots , $T[s[b]..|T|]$ is $\text{lcp}(a + 1, b)$.

Figure 1 is an example of the suffix array. Consider three substrings **ippi**, **issippi**, and **ississippi**. $\text{lcp}(2, 4) = 1$ and it means that the longest common prefix of three substrings is **i**.

| | s | lcp | |
|----|----|-----|-------------|
| 1 | 11 | - | i |
| 2 | 8 | 1 | ippi |
| 3 | 5 | 1 | issippi |
| 4 | 2 | 4 | ississippi |
| 5 | 1 | 0 | mississippi |
| 6 | 10 | 0 | pi |
| 7 | 9 | 1 | ppi |
| 8 | 7 | 0 | sippi |
| 9 | 4 | 2 | sissippi |
| 10 | 6 | 1 | ssippi |
| 11 | 3 | 3 | ssissippi |

Fig. 1. Suffix array for mississippi

The generalised suffix array for $\mathcal{U} = \{T_1, T_2, \dots, T_\ell\}$ can be built as in [4]. First we create a new string $T'' = T_1\%T_2\% \dots T_\ell$ where $\%$ is a special symbol which is smaller than any other character in Σ . Then we compute the lcp array using the technique presented in [9] in $O(|T''|)$ time. We can use only one special character $\%$ instead of ℓ different symbols: let $\%$ not match itself. We compute lcp and another ids array such that $\text{ids}[k] = i$ if $T''[k]T''[k + 1] \dots \%$ is originally a suffix of T_i (of course, $\%$ is the symbol that appears first after $T''[k]$ in T'').

Once we are given the suffix array for \mathcal{U} , a set of positive integers $\mathcal{D} = \{d_1, d_2, \dots, d_\ell\}$, and a positive integer k , a *candidate range* is a range (a, b)

which contains k distinct values in the set $\{\text{ids}[a], \text{ids}[a+1], \dots, \text{ids}[b]\}$ and each value i appears at least d_i times in the set. A *critical range* is a candidate range that does not properly contain other candidate ranges. The number of critical ranges is $O(n)$ where n is the size of the array.

Lemma 1. [14] *The answer for the generalised longest common repeat problem is $T''[s[a']]..T''[s[a'+\text{lcp}(a'+1, b')]]$ such that (a', b') is a critical range and $\text{lcp}(a'+1, b')$ is the greatest among all critical ranges.*

Proof. The proof of Lemma 1 can be found in [14].

3 Algorithms

3.1 Algorithm for the RMQ

Our algorithm is based on the simple fact.

Lemma 2. *If we have two ranges (a, b) and $(b + 1, c)$, $\text{lcp}(a, c) = \min(\text{lcp}(a, b), \text{lcp}(b + 1, c))$.*

Assume that we have an array of size n and L range minimum queries. Also we assume that the query ranges are sorted by their left end in ascending order. When they are not overlapping each other, we do not need complex algorithms: just scan the ranges and report the smallest elements. But when they are overlapping, we need to handle the overlapping ranges.

Suppose that we have three overlapping ranges (a, b) , (a', b') , and (a'', b'') such that $a < a' < a'' < b < b' < b''$ as in Fig. 2. First we can find $\text{lcp}(a, b)$ using the naïve algorithm. We define another array M and compute $M[i] = \text{lcp}(i, b)$ in $O(b - a)$ time. After computing $\text{lcp}(a, b)$, we can answer the RMQ for any range (i, b) ($a \leq i \leq b$) by looking up the value $M[i]$ in $O(1)$ time.

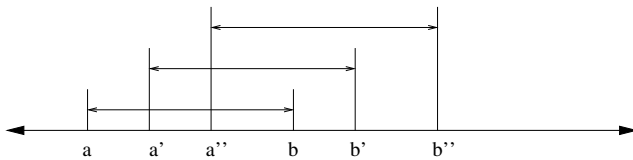


Fig. 2. An example of the RMQ for (a, b) , (a', b') , and (a'', b'')

Then we need to find $\text{lcp}(a', b')$. We know $\text{lcp}(a', b) = M[a']$ and we only have to know $\text{lcp}(b + 1, b')$. We compute $M[i] = \text{lcp}(i, b')$ ($b + 1 \leq i \leq b'$). Then we compare $M[a']$ and $M[b + 1]$. Note that we don't need to store the right end for each range in M . We can free $M[a..a' - 1]$ since it won't be used again.

Now we compute $\text{lcp}(a'', b'')$. It is $\min(\text{lcp}(a'', b), \text{lcp}(b + 1, b'), \text{lcp}(b' + 1, b''))$. As we did before, $M[i] = \text{lcp}(i, b'')$ ($b' + 1 \leq i \leq b''$). But we need to compare three values (since (a'', b'') consists of (a'', b) , $(b + 1, b')$, and $(b' + 1, b'')$) and

report the minimum. It is easy to show that in the worst case there may be L values to compare. We use a minimal heap to find the minimum. A minimal heap can be implemented by using just an array of length L . Whenever we compute $\text{lcp}(a, b)$, we store $\text{lcp}(a, b)$ which is the key for the heap and the range (a, b) . We compare $M[b' + 1] = \text{lcp}(b' + 1, b'')$ and the key of the root of the heap. If the range of the root is in (a'', b'') , we report the smaller one between $M[b' + 1]$ and the key of the root. If not, we keep on deleting the root node and fix the heap until we meet the condition.

When the new range is not overlapping the previous one, we remove all the elements in the heap. We have only to set the number of elements in the heap as 0 in $O(1)$ time. Also we can free entries in M .

The total time complexity is $O(n + L \log L)$: $O(n)$ for computing M array and $O(L \log L)$ time for answering L RMQs. The space complexity is $O(n + L)$, $O(n)$ for M and $O(L)$ for the minimum heap. Note that we use just one $O(n)$ size array: other algorithms use more than two arrays. Therefore the actual memory usage is smaller when we have a large array for lcp computation.

3.2 Repeats Finding

We explain our algorithm for repeat finding. Basically we use Lee and Pinzon Ardila’s algorithm in [14], using our new RMQ algorithm.

Step 1: We first modify each string in \mathcal{U} to consider inverted, mirror and everted repeats. For each $i = 1, 2, \dots, \ell$, we create a new string $T'_i = T_i\% \overleftarrow{T}_i\% \overline{T}_i$. And we create a string $T'' = T'_1\%T'_2\% \dots \%T'_\ell\%$.

Step 2: We build the suffix array of T'' in $O(|T''|)$ time and space, using one of [8,10,12]. We compute lcp and ids arrays also. (We may not compute ids array. Instead, we can compute $\text{ids}[k]$ from $s[k]$ in $O(\log \ell)$ time. Usually ℓ is quite small and can be considered as a constant.) This step runs in $O(|T''|)$ time and space. Note that the suffixes of T_i , \overleftarrow{T}_i and \overline{T}_i have the same value i in the ids array.

Step 3: In this step we find the critical ranges. For details, refer to [11].

Each time we keep a range (a, b) during this step. At first $a = 1$ and $b = 0$. We maintain ℓ counters c_1, c_2, \dots, c_ℓ (initially $c_1 = c_2 = \dots = c_\ell = 0$) and a counter h which contains the number of c_i ’s ($1 \leq i \leq \ell$) that are $\geq d_i$. Initially $h = 0$.

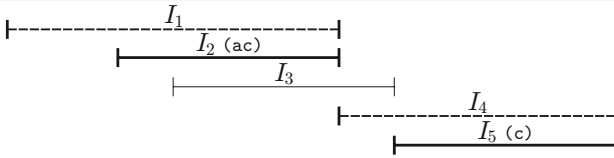
Step 3 consists of two sub-steps: *expanding* and *shrinking*. In the expanding sub-step, we find a candidate range. First we expand the range from (a, b) to $(a, b + 1)$. Then we check $\text{ids}[b]$ and set $c_{\text{ids}[b]} = c_{\text{ids}[b]} + 1$. If $c_{\text{ids}[b]} = d_i$, then $h = h + 1$. We move to the shrinking sub-step if $h = k$. Then (a, b) is a candidate range.

In the shrinking sub-step, we find a critical range from the candidate range (a, b) found in previous expanding sub-step. We start by shrinking the candidate range downwards. First, we set $c_a = c_a - 1$ and $a = a + 1$. If $c_a < d_a$, then $h = h - 1$. If $h < k$, then $(a - 1, b)$ is a critical range.

$$T_1=acac, T_2=aac, T_3=caac \quad D=\{2,1,1\}$$

$T'' = a \ c \ a \ c \ \% \ a \ a \ c \ \% \ c \ a \ a \ c \ \% \$

| | | | | | | | | | | | |
|------------|----|---|----|---|---|---|----|---|---|----|----|
| <i>i</i> | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| <i>ids</i> | 3 | 2 | 3 | 1 | 2 | 1 | 3 | 1 | 2 | 3 | 1 |
| <i>s</i> | 11 | 6 | 12 | 3 | 7 | 1 | 13 | 4 | 8 | 10 | 2 |
| <i>lcp</i> | - | 3 | 1 | 2 | 2 | 2 | 0 | 1 | 1 | 1 | 2 |
| suffix | a | a | a | a | a | a | c | c | c | c | c |
| | a | a | c | c | c | c | % | % | % | a | a |
| | c | c | % | % | % | a | | | | a | a |
| | % | % | | | | c | | | | c | c |



$$T_1'=acac\%caca\%gtgt, T_2'=aac\%caa\%gtt, T_3'=caac\%caac\%gttg$$

$T'' = a \ c \ a \ c \ \% \ c \ a \ c \ a \ \% \ g \ t \ g \ t \ \% \ a \ a \ c \ \% \ c \ a \ a \ \% \ g \ t \ t \ \% \ c \ a \ a \ c \ \% \ c \ a \ a \ c \ \% \ g \ t \ t \ g \ \% \$

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------|---|----|----|----|----|----|----|----|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| <i>i</i> | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 |
| <i>ids</i> | 1 | 2 | 2 | 2 | 3 | 3 | 2 | 3 | 1 | 3 | 1 | 1 | 2 | 3 | 1 | 3 | 1 | 2 | 3 | 3 | 2 | 6 | 4 | 13 | 11 | 24 | 38 | 14 | 26 | 40 | 12 | 25 | 39 |
| <i>suf</i> | 9 | 22 | 21 | 16 | 29 | 34 | 17 | 30 | 3 | 35 | 7 | 1 | 18 | 31 | 4 | 36 | 8 | 20 | 28 | 33 | 2 | 6 | 41 | 13 | 11 | 24 | 38 | 14 | 26 | 40 | 12 | 25 | 39 |
| <i>lcp</i> | - | 1 | 1 | 2 | 3 | 3 | 1 | 2 | 2 | 2 | 2 | 3 | 0 | 1 | 1 | 1 | 1 | 2 | 3 | 4 | 2 | 3 | 0 | 1 | 2 | 2 | 3 | 0 | 1 | 1 | 2 | 1 | 2 |
| suffix | a | a | a | a | a | a | a | a | a | a | a | a | c | c | c | c | c | c | c | c | a | a | a | a | a | a | c | a | a | a | a | a | a |
| | % | % | % | a | a | a | c | c | c | c | c | c | % | % | % | % | % | % | % | % | a | a | a | a | a | c | a | a | a | a | a | a | a |
| | | | | % | % | % | % | % | % | % | % | % | a | a | a | a | a | a | a | a | c | a | a | a | a | c | a | a | a | a | a | a | a |
| | | | | | | | | | | | | | | | | | | | | | | % | % | % | % | % | % | % | % | % | % | % | % |

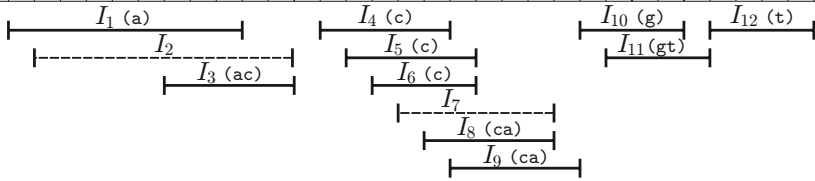


Fig. 3. Finding common repeats for $T_1 = acac$, $T_2 = aac$, and $T_3 = caac$

Once we find a critical range, we compute $\text{lcp}(a, b)$ and report the longest common prefix. We use our new RMQ algorithm here. After the RMQ, we go back to the expanding sub-step with (a, b) .

We can speed up this step although it doesn't change the time complexity. Assume that we computed $\text{lcp}(a, b)$ and are going to compute $\text{lcp}(a', b')$ such that $a < a' < b < b'$. In our new RMQ query algorithm, we find $\text{lcp}(b + 1, b')$. We are interested in the longest repeat and if $\text{lcp}(b + 1, b') < \text{lcp}(a, b)$ then it is easy to show that $\text{lcp}(a', b') < \text{lcp}(a, b)$. Therefore we know that our answer is not in (a', b') .

We include an example in Fig. 3 from [14]. We have 3 strings $T_1 = \text{acac}$, $T_2 = \text{aac}$, and $T_3 = \text{caac}$. We want to find the longest substring which appears twice in T_1 and once in T_2 and T_3 . If we do not consider inverted and mirror repeats, the above table shows that ac and c satisfy the condition. The answer is ac since it is longer than c . In this example, two critical ranges I_2 and I_5 are not overlapping. For I_2 , we compute $M[4..6] = \{2, 2, 2\}$. It follows that $\text{lcp}(I_2) = 2$. For I_5 , we free $M[4..6]$ since it is not overlapping with I_5 . We compute $M[9..11] = \{1, 1, 1\}$ and $\text{lcp}(I_5) = 1$. Unlike other algorithms for the RMQ, our algorithm does not preprocess the whole lcp array and just read 6 entries that are needed, which makes our algorithm perform better. If we consider inverted and mirror repeats, the below table shows that the answers are ac , ca , and gt . Actually one substring ac appears inverted and mirrored. I_1 and I_3 are overlapping. As we did before, we can compute $M[1..9]$ and report $\text{lcp}(I_1) = 1$. For I_3 , $M[8] = 2$ and $M[10..11] = \{2, 2\}$. By comparing two, we can find out $\text{lcp}(I_3) = 2$. For I_4, I_5, I_6, I_8 , and I_9 , we free $M[1..11]$ and perform the similar steps for $M[13..22]$ and build a minimum heap. We can safely escape I_4, I_5 , and I_6 since their lcp values are smaller than $\text{lcp}(I_3) = 2$. Then we free $M[13..22]$ and the heap and go for I_{10}, I_{11} , and I_{12} .

Theorem 1. *The generalised longest common repeat problem can be solved in $O(n + L \log L)$ time and $O(n + L)$ space, where $n = \sum_{i=1}^{\ell} |T_i|$ and L is the number of critical ranges.*

Proof. Finding critical ranges takes $O(n)$ time. Given L RMQs, our algorithm runs in $O(n + L \log L)$ time. So the total time complexity is $O(n + L \log L)$.

4 Conclusions

We proposed a simple and space-efficient algorithm for the RMQ when the queries are sorted in ascending order. We also showed that using our RMQ algorithm, it is possible to design and implement a simple algorithm for the generalised longest common repeat problem.

Future works include handling the case where queries are not ordered.

References

1. Bender, M. A., Farach-Colton, M.: The LCA problem revisited. In Proceedings of the Fourth Latin American Symposium (2000) 88–94
2. Bender, M. A., Farach-Colton, M., Pemmasani, G., Skiena, S., Sumazin, P.: Lowest common ancestors in trees and directed acyclic graphs. *Journal of Algorithms* **57**(2) (2005) 75–94
3. Berkman, O., Vishkin, U.: Recursive star-tree parallel data structure *SIAM Journal on Computing* **22**(2) (1993)
4. Dori, S., Landau, G. M.: Construction of Aho-Corasick automaton in linear time for integer alphabets. In Proceedings of the 16th Annual Symposium on Combinatorial Pattern Matching (CPM 2005), (2005) 168–177

5. Ferragina, P., Manzini, G.: Opportunistic data structures with applications. In FOCS (2000) 390-398
6. Fischer, J., Heun, V.: Theoretical and Practical Improvements on the RMQ-Problem, with Applications to LCA and LCE. In Proceedings of the 17th Annual Symposium on Combinatorial Pattern Matching (CPM 2006), (2006) 36–48
7. Harel, D., Tarjan, R. E.: Fast algorithms for finding nearest common ancestor. SIAM Journal on Computing **13**(2) (1984) 338–335
8. Kärkkäinen, J., Sanders, P.: Simpler linear work suffix array construction. In Proceedings of the 13th International Colloquium on Automata, Languages and Programming (ICALP 2003), (2003) 943–945
9. Kasai, T., Lee, G., Arimura, H., Arikawa, S., Park, K.: Linear-time longest-common-prefix computation in suffix arrays and its applications. In Proceedings of the 12th Annual Symposium on Combinatorial Pattern Matching (CPM 2001), (2001) 181–192
10. Kim, D. K., Sim, J. S., Park, H., Park, K.: Linear-time construction of suffix arrays. In Proceedings of the 14th Annual Symposium on Combinatorial Pattern Matching (CPM 2003) (2003) 186–199
11. Kim, S.-R., Lee, I., Park, K.: A fast algorithm for the generalised k -keyword proximity problem given keyword offsets. Information Processing Letters **91**(3) (2004) 115–120
12. Ko, P. Aluru, S.: Space-efficient linear time construction of suffix arrays. In Proceedings of the 14th Annual Symposium on Combinatorial Pattern Matching (CPM 2003), (2003) 200–210
13. Lee, I., Iliopoulos, C. S., Park, K.: Linear time algorithm for the longest common repeat problem. In Proceedings of the 11th String Processing and Information Retrieval (SPIRE 2004), (2004) 10–17
14. Lee, I., Pinzon Ardila, Y. J.: Linear time algorithm for the generalised longest common repeat problem. In Proceedings of the 12th String Processing and Information Retrieval (SPIRE 2005), (2005) 191–201
15. Schieber, B. Vishkin, U.: On finding lowest common ancestor: Simplification and parallelization. SIAM Journal on Computing **17**(6) 1253–1262

Improved Bacterial Foraging Algorithms and Their Applications to Job Shop Scheduling Problems

Chunguo Wu¹, Na Zhang¹, Jingqing Jiang^{1,2}, Jinhui Yang¹, and Yanchun Liang¹

¹ College of Computer Science and Technology, Jilin University, Key Laboratory of Symbol Computation and Knowledge Engineering of the Ministry of Education, Changchun 130012, P.R. China

² College of Mathematics and Computer Science, Inner Mongolia University for Nationalities, Tongliao 028043, China
ycliang@jlu.edu.cn

Abstract. Bacterial foraging algorithm is a novel evolutionary computation algorithm proposed four years ago, which is based on the foraging behavior of *E.coli* bacteria living in human intestine. In this paper an improved operation, individual-based search, is presented with regard to the important component (Chemotaxi) of bacterial foraging algorithm. The improved algorithm is applied to job shop scheduling benchmark problems. Numerical experiments show the effectiveness of the improved algorithm.

1 Introduction

In the last forty years, researchers have been trying to simulate the biological systems from various aspects and proposed some effective bionic algorithms, including artificial neural network (ANN), genetic algorithm (GA), ant colony optimization (ACO), particle swarm optimization (PSO) and artificial immune system (AIS), etc. These bionic algorithms provide novel paradigms for engineering problems by mimic the specific structures or behaviors of certain creatures.

Bacterial foraging optimization (BFO) is a quite young but effective bionic algorithm. It was presented by Passino in *IEEE Control Systems Magazine* in Ref. 2 (2002) and later in the same year Ref. 3 was published in the *Journal of Optimization Theory and Applications* for demonstrating it a simple but powerful optimization tool¹. BFO is essentially a random search algorithm, which mimics the foraging behavior of *E. coli* bacteria. After its appearance, BFO has been applied successfully to some engineering problems. By using BFO, Ref. 4 dealt with harmonic estimation of signal distorted with additive noise and Ref. 5 designed PID controller for a multivariable system. However, compared with other bionic algorithms, BFO just starts its life and only obtains limited applications. Its powerful potential is expected to emit in further improvements and applications.

¹ The MATLAB code of the algorithm and the tri-dimensional functions experimented, can also be found on the web address of a recent book from the same author (*Biomimicry for Optimization, Control and Automation*, Springer-Verlag, London, UK, 2005), at http://www.ece.osu.edu/~passino/ICbook/ic_index.html.

2 Brief Review of Bacterial Foraging Optimization

E. coli bacteria exist in intestines of most animals on the earth. *E. coli* bacterium has a control system, which directs its behaviors in food foraging. The foraging process consists of a series moves towards food sources. The control system is in charge of evaluating changes from one state to the other states to provide reference information for *E. coli* bacterium's next state change. A change between two states is called as a move and the states include advancing direction and step length. *E. coli* bacteria approach gradually their food sources under the influence of its control system. Biological studies show that the foraging process includes the below four steps: (1) search for a possible food region, (2) decide to whether or not enter into the possible food region, (3) perform a careful search if it enters into a new region, (4) decide to either keep stay in the current region or emigrate into a new and more ideal region, after they consume some food in the current region. In general, if the bacteria are trapped into a region in deficiency of food, they might draw a conclusion based on past experience that other regions must be in abundance of food. Due to this conclusion, bacteria would change their states. Hence, each decision of state change is made under the physiological and environmental constraints with the final aim to maximize the obtained energy in unit time.

To mimic the aforementioned biological principles shown in the foraging behavior of *E. coli* bacteria, Kevin M. Passino in 2000 presented BFO for distributed optimization and control 2. Similar with most swarm intelligence-based random search algorithms, by using BFO to solve optimization problems, the first step is to encode the solution; and then manipulate the individuals based on the full usage of swarm information; at last reach the optimal (or swarm optimal) solution. An optimization period of BFO consists of three events: chemotactic event, duplicate event and elimination-dispersal event.

These events are illustrated as follows:

Chemotactic event: Microbiological studies show that *E. coli* bacteria move by their flagella. Hence, biologists call the flagella of *E. coli* bacteria as biological engines 6. When all the flagella sway counterclockwise, the *E. coli* bacteria move forward; and when all the flagella sway clockwise, the *E. coli* bacteria slow down and tumble in its place. The foraging of *E. coli* bacteria is accompanied by the alteration of the last two behaviors.

In the foraging of bacteria, the swaying corresponds to the evaluation of individual's current environment, and then decides to whether or not adjust the current position and the parameters (e.g., direction of the next move and the step length) regarding to how to adjust its position. In BFO, the formula of direction change is given as follows:

$$\theta^i(j+1, k, l) = \theta^i(j, k, l) + C(i)\phi(j) \quad (1)$$

where $\theta^i(j, k, l)$ stands for the current position of the i th individual; j , k and l indicate for the numbers of chemotactic events, duplicate iterations and elimination-dispersal events, respectively; $\phi(j)$ stands for the new advancing direction decided by flagella swaying and $C(i)$ for the step length.

Duplicate event: After a period of food search, the foraging strategies of some bacteria appear inferior evidently. These bacteria with inferior foraging strategies suffer a high probability to be removed out of the population due to their low ability to find enough food. To keep the population size constant, a portion of bacteria with superior foraging strategies are duplicated to take the places of removed ones.

The duplicate event is fulfilled as follows: Let the population size is S , the number of bacteria to be removed is defined as

$$S_r = S / 2 \quad (2)$$

Firstly, rank all of the individuals regarding to evaluations of their positions, and then remove out the last half (S_r) individuals and duplicate one copy for each of the residual half (S_r) ones to keep a constant population size.

Elimination-dispersal event: The changes of the environment where the bacteria population live, for example, the sudden increasing of the temperature, the flushing with water and the affection by other entities, will affect the behavior of the bacteria heavily. All of these factors possibly cause gradual or sudden changes of the population. The population changes might include that all bacteria in the current region are killed or part of them move to a new region. The elimination-dispersal event is an evolution operation designed to imitate this biological process. This operation likely destroys the performance of the chemotactic event, but also likely promotes it, since dispersal might place bacteria near a better food source.

The elimination-dispersal event is triggered with probability P_{ed} . If certain individual satisfies the dispersal condition, it should be deleted and then a new individual should be generated. This operation means that the individual moves to a new position.

The procedure of solving optimization problem with bacterial foraging includes: (1) encoding the solution of the problem, (2) designing the evaluation function, (3) generating the initial population, (4) optimizing the objective function by the interaction of individuals. The steps of the algorithm are as follows:

Pseudocode of bacterial foraging algorithm

- Step 1. Initialize the population
 - Step 2. Evaluate the individual using evaluation function
 - Step 3. Three loops of optimization
 - Inner loop: chemotactic event;
 - Middle loop: duplicate event;
 - Outer loop: elimination-dispersal event
 - Step 4. Decode the optimal individual to obtain the final solution
-

There are three loops of optimization in the algorithm. The outer loop is elimination-dispersal event, the middle loop is duplicate event and the inner loop is chemotactic event. The inner loop, chemotactic event, is the core of the three loops. It corresponds

to the direction selection scheme which is the central step employed by a living creature to search food and in charge of the decisions that whether or not enter into a new region, how long does the individual stay in the current region, which direction should be selected in the next move. These decisions mean that the chemotactic event has important influence to the algorithm convergence.

3 The Improved BFO Algorithm

3.1 Encoding Scheme

The improved bacterial foraging algorithm will be applied to solving the job shop scheduling problems (JSSP). Hence, we design the encoding scheme taking the consideration of JSSP. The numbers in a code are the numbers of the workpieces and the i th ($i=1, 2, \dots, m$) appearance of the number j ($j=1, 2, \dots, n$) denotes the i th operation of the j th workpiece. Fig.1 shows an example of our encoding scheme. Fig. 2 and Fig. 3 are the sequence and the Gantt chart corresponding to the Fig. 1. In Fig. 2 the group (i, j, k, t) means that the j th operation of the i th workpiece requires the t time span on the k th machine.

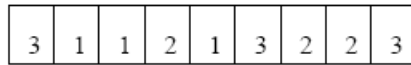


Fig. 1. Example of our coding scheme

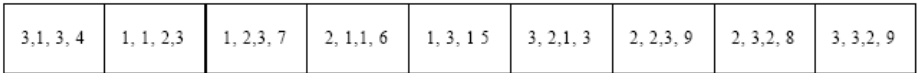


Fig. 2. The decoded operation sequence corresponding the code in Fig.1

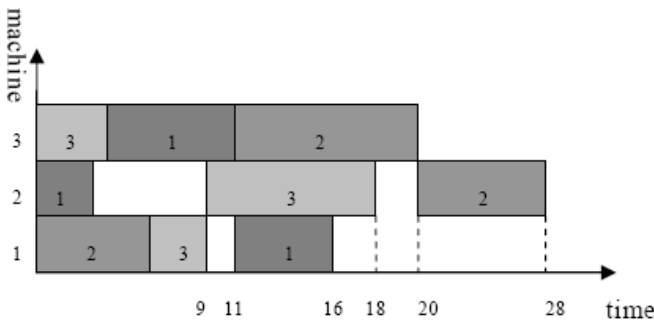


Fig. 3. Chart corresponding to the code in Fig.1

3.2 The Improved Chemotactic Event

There are two foraging methods in the biological foraging search scheme: patrol and ambush. The foragers search food by changing their position continually in the patrol scheme, while they keep the static state and wait for the preys passing from them in the ambush scheme. In fact, the foraging method of most animals alternates between the patrol and the ambush, which is called bound search. *E.coli* bacteria just search food by this bound scheme, i.e., the patrol and ambush are applied alternately. When the search scheme changes from ambush to patrol, the foragers usually adjust the patrol direction according to the information accumulated in the previous experience. When they change from patrol to ambush, the foragers decide the next patrol direction according to the environment.

Inspired by the intelligent action which embodied in the process of searching for food, an improved search schemes for chemotactic event is proposed in this paper. The briefly descriptions of the improved schemes is as follows

In the process of searching food, each bacterium searches its possible direction. Searching based on individual means that the individual adjust the patrol direction according to its own information. The information is accumulated historically during the food searching process.

The design of this search scheme is as follows: firstly, select an individual from the population and randomly select two positions on the encoding sequence. The codes between these two positions are called stable region. The codes in the stable region will not change at the following operation. So an individual is divided into three parts generally. Except for the stable region, the regions in the two sides are called exchanging interval A and exchanging interval B, respectively. And then exchange in these two intervals, i.e., randomly ranking the sequences in interval A and interval B, respectively. This exchanging could guarantee that the encoding sequence is a valid solution.

The individual moves forward into the direction selected after the exchanging. Then evaluate the current state. If the new individual is near the food source, the previous exchanging is correct and the new individual will take the place of the old one. On the contrary, the individual moves back to its previous position. Fig. 4 shows the flow chart of the algorithm. The variables in Fig. 4 are described as follows: x denotes the selected individual, A and B denote the left side and the right side of the stable region, respectively.

4 Simulation Experiments and Results

In this section, the proposed improved algorithm is applied to job shop schedule problem to test its efficiency.

The original BFO algorithm and the improved BFO algorithm are implemented using C language. The results are showed in Table 1. The test data come form the LA test problems generated by Lawrence in 1984 [7]. BKS is the current optimal solution. IBFO is the improved algorithm and BFO is the standard bacterial foraging algorithm. TS and BS are the results of taboo search and beam search algorithm reported in References 8 and 9.

It can be seen from Table 1 that the proposed algorithm in this paper could find the optimization for most cases. Fig. 5 shows the makespans trend with regard to the iteration number, where the solid, dashed and dotted lines denote the maximum, average and minimum values, respectively.

Table 1. The results of experiments

| Instance | Size | BKS | BFO | IBFO | TS | BS |
|----------|------|-----|-----|------|-----|-----|
| LA01 | 10*5 | 666 | 698 | 666 | 666 | 666 |
| LA02 | 10*5 | 655 | 740 | 668 | 655 | 704 |
| LA03 | 10*5 | 597 | 671 | 617 | 603 | 650 |
| LA04 | 10*5 | 590 | 657 | 604 | 590 | 620 |
| LA05 | 10*5 | 593 | 593 | 593 | 593 | 593 |
| LA06 | 15*5 | 926 | 930 | 926 | 926 | 926 |
| LA07 | 15*5 | 890 | 952 | 890 | 890 | 890 |
| LA08 | 15*5 | 863 | 898 | 863 | 863 | 863 |
| LA09 | 15*5 | 951 | 993 | 951 | 951 | 951 |
| LA10 | 15*5 | 958 | 962 | 958 | 958 | 958 |

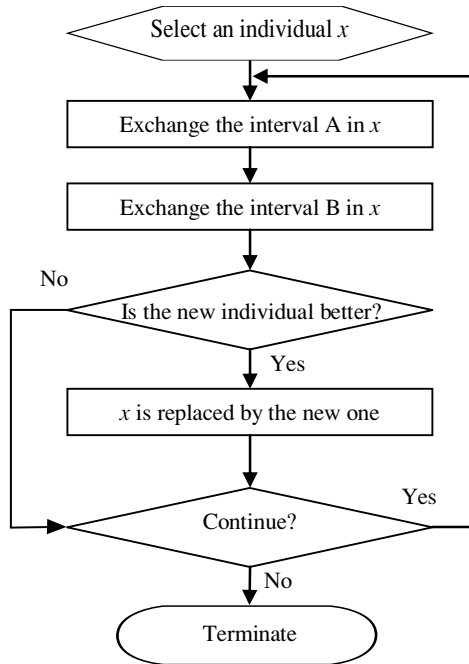


Fig. 4. Chart of the individual-based search scheme

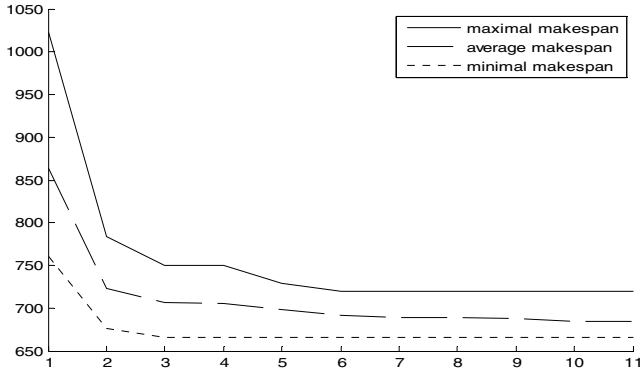


Fig. 5. Makespan trend for LA01 corresponding to iterations

5 Conclusions

In this paper we analyze the direction selection scheme for bacterial foraging and propose an improved chemotactic event named individual-based searching scheme. The proposed method is verified by solving job shop scheduling problems and competitive results are obtained.

Acknowledgement. The authors are grateful to the support of the National Natural Science Foundation of China (60433020, 60673023), the science-technology development project of Jilin Province of China (20050705-2), the European Commission under grant No. TH/Asia Link/010 (111084), the graduate innovation lab of Jilin University (503043), and “985” project of Jilin University of China.

References

1. Ramos, V., Fernandes, C., Rosa, A. C.: On Ants, Bacteria and Dynamic Environments.
2. Passino, K. M.: Biomimicry of Bacterial Foraging for Distributed Optimization and Control. *IEEE Control System Magazine*, (2002) 52-67.
3. Liu, Y., Passino, K.M.: Biomimicry of Social Foraging Bacteria for Distributed Optimization: Models, Principles, and Emergent Behaviors. *Journal of Optimization Theory and Applications*, (2002) 115(3): 603-628.
4. Mishra, S.: A hybrid least square-fuzzy bacterial foraging strategy for harmonic estimation. *IEEE Trans. Evolutionary Computation*. (2005) 9(1): 61-73.
5. Kim, D. H., Cho, J. H.: Adaptive Tuning of PID Controller for Multivariable System Using Bacterial Foraging Based Optimization. *AWIC 2005, LNAE 3528*, (2005): 231-235.
6. Berg, H.: Motile behavior of bacteria. *Phys. Today*, (2000): 24-29.
7. Wang, L.: Job Shop Scheduling and Its Genetic Algorithm Solution. Tsinghua Publishing Company, 2003 (in Chinese).

8. Xi, W., Tan, X. B., and Baras, J. S.: A Hybrid Scheme for Distributed Control of Autonomous. American Control Conference June 8-10, 2005.
9. Baras, J. S., Tan, X. B. and Hovareshti, P.: Decentralized Control of Autonomous vehicles. Proceedings of the 42ed IEEE Conference on Decision and Control. Maui, Hawaii, USA, December, 2003.

An Evolutionary Approach for Approximating the Solutions of Systems of Linear Fuzzy Equations*

Nguyen Hoang Viet and Michał Kleiber

Institute of Fundamental Technological Research, Polish Academy of Sciences,
Swietokrzyska 21, 00-049 Warsaw, Poland
{viet,mkleiber}@ippt.gov.pl

Abstract. In this paper systems of linear equations $Ax = b$, where both A and b contain uncertain factors in terms of fuzziness are investigated. The classical solutions being vectors of fuzzy numbers are considered. The complex problem of finding the exact classical solutions is replaced by a corresponding optimization task with the cost function based on the Hausdorff metric. This cost function is next minimized with use of genetic algorithms. A number of numerical experiments are provided in order to verify the given approach. The results and some conclusions are also included.

1 Introduction

Several real world problems in economics, finance, mechanics etc. can lead to solving a system of linear equations. When the coefficients of the system are imprecise, it may be convenient to use *interval real numbers* to express that impreciseness [1]. Hence, one obtains a system of interval linear equations, which can be solved using several techniques [8]. If however one has additional knowledge about the parameters of the system and if different degrees of belief can be assigned to different values of the parameters, then instead of interval numbers, one can consider using *fuzzy real numbers*.

A system of linear equations $Ax = b$, where some (or all) of the elements a_{ij} of the matrix A and b_i of the vector b are fuzzy numbers is called *System of Linear Fuzzy Equations*, or sometimes *Fuzzy Linear System*. For sake of simplicity we hereby denote systems of linear fuzzy equations by SLFE. Many variants of SLFE have recently been investigated by researchers. In [4], a special type of SLFE where A is a crisp matrix and b is a vector of fuzzy numbers is considered. The original $n \times n$ system is replaced by another $(2n) \times (2n)$ system $Sy = b'$, where S is a $(2n) \times (2n)$ crisp matrix obtained from the elements of A , b' is a crisp vector of the (parameterized) endpoints of the α -cuts of b and y is a crisp vector representing the endpoints of the α -cuts of the unknowns. In [9] an iteration method based on interval arithmetics for solving the fuzzy linear

* This work is supported by the grant no. N519 020 31/3900 from the Polish Ministry of Science and Higher Education (Fund for Scientific Research in 2006-2007).

systems $x = Ax + U$, where A is a crisp matrix and U is a vector of fuzzy numbers is given. It is shown that if the condition $\|A\|_\infty < 1$ holds then it is possible to construct a linear mapping with a unique fix-point being the solution of the given system.

Another type of SLFE is discussed in [5]. Here both matrix A and vector b contain fuzzy number elements, however crisp solutions are of interest. Such solutions are approximated by applying a simple fuzzy neural network. The network has one single output neuron which is fully connected with all input neurons. The (crisp) connection weights of this neural net represent an approximated solution of SLFE.

The most interesting case is when A , b and the vector of unknowns x contain fuzzy elements. Since now we will emphasize that by placing a tilde over them, i.e. \tilde{A} , \tilde{b} and \tilde{x} . In [3], the systems $\tilde{A}\tilde{x} = \tilde{b}$ with triangular fuzzy numbers are studied. Several definitions of a solution for such systems are given, including that based on the united solution of systems of interval linear equations [8], as well as on the Cramer rule etc. The study is however focused rather on the theoretical than the computational aspect of the problem.

In this paper we will limit ourself on the *classical solutions* of $\tilde{A}\tilde{x} = \tilde{b}$, as mentioned in [3]. A vector of fuzzy numbers \tilde{x}^* is called the solution of the linear fuzzy system $\tilde{A}\tilde{x} = \tilde{b}$ if $\tilde{A}\tilde{x}^* = \tilde{b}$ in the sense of regular fuzzy arithmetics. Most attention will be paid on the computational aspect of the problem. In section [2] various basic notions are given. Some details on the classical solution are discussed in section [3]. Next, an evolutionary approach to approximate the classical solutions is proposed in section [4]. The results of various numerical tests are given in section [5], whereas in section [6] some final conclusions are drawn.

2 Preliminaries

The concept of fuzzy sets was first introduced by Zadeh in [11]. Fuzzy sets can be used, among other things, to represent inexact data, to express vague concepts, rules etc. A fuzzy set \tilde{A} over a universe \mathcal{X} is characterized by a mapping $\mu_{\tilde{A}}$ from \mathcal{X} into $[0, 1]$. This mapping is called the membership function of \tilde{A} . For all $x \in \mathcal{X}$, $\mu_{\tilde{A}}(x)$ is the degree of membership of x in the fuzzy set \tilde{A} .

For all $\alpha \in [0, 1]$, the subset $\tilde{A}(\alpha) = \{x \in \mathcal{X} : \mu_{\tilde{A}}(x) \geq \alpha\}$ of \mathcal{X} is referred to as the (weak) α -cut of \tilde{A} . In this paper, *fuzzy real numbers*, or shortly fuzzy numbers are of major concern. We hereby denote a fuzzy number by placing a tilde over lower case characters, such as \tilde{a} , \tilde{b} etc. A fuzzy number \tilde{a} is a fuzzy set over the set of all real numbers \mathbb{R} where:

- \tilde{a} is convex, i.e. all α -cuts $\tilde{a}(\alpha)$ of \tilde{a} are convex subsets of \mathbb{R} (real intervals).
- there exist exactly one value $a \in \mathbb{R}$ for which $\mu_{\tilde{a}}(a) = 1$. This value is referred to as the *mode* of the fuzzy number \tilde{a} .

In other words, fuzzy numbers are convex, normalized and unimodal fuzzy subsets of the real line \mathbb{R} . So defined fuzzy real number \tilde{a} can usually be interpreted

as *about* a , where the corresponding uncertainty of this expression is characterized by the membership function $\mu_{\tilde{a}}$. The set of all fuzzy real numbers is denoted hereby as $\mathcal{F}(\mathbb{R})$.

Let $\mathcal{K}(\mathbb{R})$ be the family of all closed subsets of \mathbb{R} . The *Hausdorff metric* on $\mathcal{K}(\mathbb{R})$ is defined as follows:

$$H(A, B) = \max \left\{ \sup_{a \in A} \inf_{b \in B} |a - b|, \sup_{b \in B} \inf_{a \in A} |a - b| \right\}, \tag{1}$$

for all $A, B \in \mathcal{K}(\mathbb{R})$. This metric can be generalized into that on $\mathcal{F}(\mathbb{R})$, namely:

$$\mathbf{H}_\infty(\tilde{a}, \tilde{b}) = \sup_{\alpha \in (0,1]} H(\tilde{a}(\alpha), \tilde{b}(\alpha)), \tag{2}$$

for all $\tilde{a}, \tilde{b} \in \mathcal{F}(\mathbb{R})$. Since for all $\alpha \in (0, 1]$, $\tilde{a}(\alpha)$ and $\tilde{b}(\alpha)$ are interval numbers, \mathbf{H}_∞ can be rewritten as follows:

$$\mathbf{H}_\infty(\tilde{a}, \tilde{b}) = \sup_{\alpha \in (0,1]} \max \left\{ \left| \tilde{a}(\alpha)^L - \tilde{b}(\alpha)^L \right|, \left| \tilde{a}(\alpha)^R - \tilde{b}(\alpha)^R \right| \right\}, \tag{3}$$

where the indices L and R indicate the left and right endpoints of interval real numbers [1].

Standard arithmetic operations on fuzzy numbers can be defined with use of Zadeh’s *extension principle* [12]. Let \tilde{a} and \tilde{b} be fuzzy numbers, let \diamond be one of the operators $\{+, -, \times, /\}$. Additionally, assume that the support $\text{supp}(\tilde{b})$ of \tilde{b} ($\text{supp}(\tilde{b}) = \tilde{b}(0)$) does not contain zero in the case of division. Let’s define a mapping $\mu_{\tilde{c}} : \mathbb{R} \rightarrow [0, 1]$ as follows:

$$\forall z \in \mathbb{R}, \mu_{\tilde{c}}(z) = \sup_{z=x \diamond y} \min \{ \mu_{\tilde{a}}(x), \mu_{\tilde{b}}(y) \}. \tag{4}$$

It can be shown that the fuzzy set \tilde{c} with the membership function $\mu_{\tilde{c}}$ defined above is also a fuzzy number. This fuzzy number is referred to as the sum, difference, product or quotient of \tilde{a} and \tilde{b} for \diamond being $+$, $-$, \times and $/$ respectively.

In practice however fuzzy computation usually makes use of the α -cuts. One can easily show that for $\tilde{a}, \tilde{b} \in \mathcal{F}(\mathbb{R})$, $(\tilde{a} \diamond \tilde{b})(\alpha) = \tilde{a}(\alpha) \diamond \tilde{b}(\alpha)$, where on the right side of this equation the operator \diamond refers to that of interval arithmetics [1].

3 Systems of Linear Fuzzy Equations

Let us now consider the following system of linear fuzzy equations:

$$\tilde{A}\tilde{x} = \tilde{b} \tag{5}$$

where $\tilde{A} = [\tilde{a}_{ij}]^{m \times n}$ is an $m \times n$ matrix of fuzzy numbers and $\tilde{b} = [\tilde{b}_i]^{m \times 1}$ is a vector of fuzzy numbers. In general, m can be different from n . A vector of

fuzzy numbers $\tilde{x}^* = [\tilde{x}_i^*]^{n \times 1}$ is called the solution of (5) if by substituting \tilde{x}^* into the left side of (5) and performing standard fuzzy number multiplication and addition, one obtains the right side vector \tilde{b} . In other words:

$$\sum_{j=1}^n \tilde{a}_{ij} \tilde{x}_j = \tilde{b}_i, \forall i = \overline{1, m}. \tag{6}$$

A standard way to solve this system is to replace it by a set of systems of interval linear equations (or a parameterized interval system) [3]:

$$\sum_{j=1}^n [\tilde{a}_{ij}(\alpha)^L, \tilde{a}_{ij}(\alpha)^R] [\tilde{x}_j(\alpha)^L, \tilde{x}_j(\alpha)^R] = [\tilde{b}_i(\alpha)^L, \tilde{b}_i(\alpha)^R], \forall i \in \overline{1, m}. \tag{7}$$

This approach however has some serious drawbacks. First, for a given $\alpha \in [0, 1]$ finding the interval solution of (7) is a problem of exponential complexity. Second, assuming that for each $\alpha \in [0, 1]$ the solution $\tilde{x}_j(\alpha)$ of (7) exists and can be found, $\{\tilde{x}_j(\alpha) : \alpha \in [0, 1]\}$ need not be a family of nested intervals. This means that it need not define a proper fuzzy number.

One of the possibilities to overcome these drawbacks is to apply the GA-based approach to approximate the solution of systems of interval linear equations as proposed in one of our previous papers [10]. Starting with $\alpha = 0$ one can find the approximating solution $[\tilde{x}_j(0)^L, \tilde{x}_j(0)^R]$, $j = \overline{1, n}$ of (7). Having the solution $[\tilde{x}_j(\alpha_k)^L, \tilde{x}_j(\alpha_k)^R]$ for some $\alpha_k \in [0, 1)$, one can again apply the algorithm to compute $[\tilde{x}_j(\alpha_l)^L, \tilde{x}_j(\alpha_l)^R]$ for some $\alpha_l \in (\alpha_k, 1]$ with the constraint $\tilde{x}_j(\alpha_l) \subseteq \tilde{x}_j(\alpha_k)$. This procedure ensures that for all $j = \overline{1, n}$, $\{\tilde{x}_j(\alpha) : \alpha \in [0, 1]\}$ represents a proper fuzzy number.

We will propose here however a direct technique to approximate the classical solution of (5), which can be seen as an extension of that provided in [10] to the fuzzy case. The details of this approach will be discussed in the next section.

4 Genetic Algorithm for Solving SLFE

Let us now again consider the SLFE given in (6). Let $\tilde{x} = [\tilde{x}_j]$ be an arbitrary vector of fuzzy numbers. Assume that by applying standard fuzzy number multiplication and addition one obtains $\sum_{j=1}^n \tilde{a}_{ij} \tilde{x}_j = \tilde{\gamma}_i, \forall i = \overline{1, m}$. Let us now define a function over the family of all vector of fuzzy numbers in $\mathcal{F}(\mathbb{R})^m$ as follows:

$$f(\tilde{x}) = \sum_{i=1}^m \mathbf{H}_\infty(\tilde{\gamma}_i, \tilde{b}_i) = \sum_{i=1}^m \mathbf{H}_\infty\left(\sum_{j=1}^n \tilde{a}_{ij} \tilde{x}_j, \tilde{b}_i\right), \tag{8}$$

where \mathbf{H}_∞ is the extended Hausdorff metric defined in (3). It can be observed that if \tilde{x}^* is the exact solution of (5) then it is also the minimum of f . Hence one can solve (5) by minimizing the cost function f .

Discretizing fuzzy numbers at p levels of confidence $\alpha_1, \alpha_2, \dots, \alpha_p$ where $p > 1$ and $0 = \alpha_1 < \alpha_2 < \dots < \alpha_p = 1$, the cost function becomes:

$$F(\tilde{x}_1(\alpha_{1 \rightarrow p}), \dots, \tilde{x}_n(\alpha_{1 \rightarrow p})) = \sum_{i=1}^m \max_{k=\overline{1,p}, \psi=\{L,R\}} \left| \tilde{\gamma}_i(\alpha_k)^\psi - \tilde{b}_i(\alpha_k)^\psi \right|. \quad (9)$$

Here by $\tilde{x}(\alpha_{1 \rightarrow p})$ we denote the discretized representation of the fuzzy number \tilde{x} . While optimizing the cost function F , one has to take into account the condition on the $\tilde{x}_i(\alpha_k)$, that is for all $r, s = \overline{1,p}$ and $r < s$, $\tilde{x}_i(\alpha_s) \subseteq \tilde{x}_i(\alpha_r)$. As it can be observed, F is not everywhere differentiable and gradient based optimization techniques may not applicable. In this paper, a steady state genetic algorithm is designed to minimize the cost function F .

4.1 The Steady State Genetic Algorithm

The steady state genetic algorithm starts with a population of randomly generated individuals (genomes). At each generation it creates a temporary set of new individuals by performing genetic operations (i.e. selection, crossover and mutation) on current individuals. The offsprings are then added to the population and their fitness values are computed. Next, all individuals in the extended population are sorted with respect to their fitness values and the worst individuals are removed. Denote by $P(t)$ the population at step t , by S the number of individuals in the population. Let M be the number of new individuals additionally created at each iteration ($M < S$). Let T be the maximal number of iterations. The steady state strategy is explained in Fig. 1.

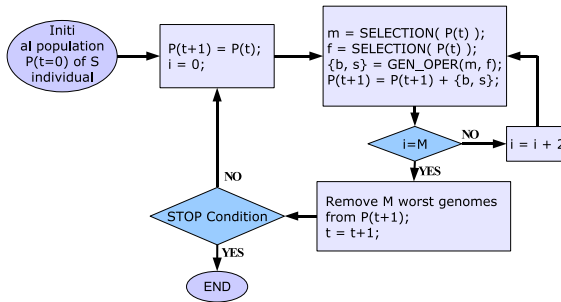


Fig. 1. The steady state strategy

4.2 Genetic Operators

In order to be able to explore the GA in solving the SLFE problem, the genomes must be encoded so that each of them is a proper representation of some potential solution of (5). Namely, each genome is a vector of discretized fuzzy numbers, i.e. $\bar{x} = (\tilde{x}_1(\alpha_{1 \rightarrow p}), \dots, \tilde{x}_n(\alpha_{1 \rightarrow p}))^T$. Here the notation \bar{x} is used instead of \tilde{x} to indicate that \bar{x} is the corresponding discretized version of \tilde{x} .

Selection. Various selection schemes such as roulette wheel selection, tournament selection etc. can be used. In each case, selection is made based on the fitness values of genomes. Here the fitness value of each genome is computed from its associating cost function value (9) and then by applying a linear scaling scheme.

Crossover. The aim of crossing over during the evolution process is to allow genetic material to be exchanged between individuals. The following arithmetic mating scheme is applied: let $\bar{x} = (\tilde{x}_1(\alpha_{1 \rightarrow p}), \dots, \tilde{x}_n(\alpha_{1 \rightarrow p}))^T$ and $\bar{y} = (\tilde{y}_1(\alpha_{1 \rightarrow p}), \dots, \tilde{y}_n(\alpha_{1 \rightarrow p}))^T$ be two genomes selected for crossing over. First, a random real number $\epsilon \in (0, 1)$ is chosen. Next, two new offsprings $\bar{b} = (\tilde{b}_1(\alpha_{1 \rightarrow p}), \dots, \tilde{b}_n(\alpha_{1 \rightarrow p}))^T$ and $\bar{s} = (\tilde{s}_1(\alpha_{1 \rightarrow p}), \dots, \tilde{s}_n(\alpha_{1 \rightarrow p}))^T$ are generated using interval number arithmetics:

$$\begin{aligned} \tilde{b}_i(\alpha_k) &= p.\tilde{x}_i(\alpha_k) + (1 - p).\tilde{y}_i(\alpha_k) \\ \tilde{s}_i(\alpha_k) &= (1 - p).\tilde{x}_i(\alpha_k) + p.\tilde{y}_i(\alpha_k), \end{aligned}$$

for $i = \overline{1, n}$ and $k = \overline{1, p}$. It can be observed that both $\tilde{b}_i(\alpha_{1 \rightarrow p})$ and $\tilde{s}_i(\alpha_{1 \rightarrow p})$ represent a proper fuzzy number.

Mutation. The aim of mutation is to maintain the diversity of genomes from generation to generation. In the algorithm presented here, an extended version of nonuniform mutation is applied. Namely let $\Delta(t, r)$ ($t \in \mathbb{N}$ and $r \in \mathbb{R}^+$) be a pseudo-random number generator with the following properties:

1. $\Delta(t, r)$ generates pseudo-random numbers between 0 and r ,
2. $\forall \epsilon > 0, \forall t_1, t_2 \in \mathbb{R}^+ : t_1 < t_2, \text{Prob}\{\Delta(t_2, r) < \epsilon\} > \text{Prob}\{\Delta(t_1, r) < \epsilon\}$. In other words, the probability of $\Delta(t, r)$ to produce a value close to 0 increases when t increases.

In this paper $\Delta(t, r) = r(1 - \omega^{1-t/T})$ is used. Here T is the maximal number of populations and ω is a pseudo-random number generator of uniform distribution in $[0, 1]$.

Let the i -th component $\tilde{x}_i(\alpha_{1 \rightarrow p})$ of a genome \bar{x} in the t -th population be chosen for mutation. Starting from $k = 1$ up to $k = p$, each interval $\tilde{x}_i(\alpha_k)$ is shifted by an amount λ_{ik} to $\tilde{x}_i'(\alpha_k)$, i.e. $\tilde{x}_i'(\alpha_k) = \tilde{x}_i(\alpha_k) + \lambda_{ik}$, where:

$$\lambda_{ik} = -\Delta\left(t, \tilde{x}_i(\alpha_k)^L - \tilde{x}_i(\alpha_{k-1})^L\right) + \lambda_{i,k-1}$$

or

$$\lambda_{ik} = \Delta\left(t, \tilde{x}_i(\alpha_{k-1})^R - \tilde{x}_i(\alpha_k)^R\right) + \lambda_{i,k-1},$$

$\tilde{x}_i'(\alpha_0)$ is the largest interval containing the support of \tilde{x}_i and $\lambda_{i,0} = 0$. This mutation scheme allows the GA to search for the solution in the whole input space at the beginning phase, whereas as evolution goes on, this process becomes local. It also ensures that the mutated genomes still represent vectors of proper fuzzy numbers.

5 Experimental Results

The proposed GA technique was applied to approximate the classical solutions of various systems of linear fuzzy equations with different values of m and n . For each configuration of $m \times n$, a set of 20 different pairs (\tilde{A}, \tilde{x}^*) were randomly generated. Each element of \tilde{A} as well as of \tilde{x}^* is a fuzzy real number with the support in $[-2, 2]$. Next, the right hand vectors \tilde{b} were computed according to $\tilde{b} = \tilde{A}\tilde{x}^*$ using standard fuzzy arithmetics.

The size of the population S , the maximal number of iterations T , the probability of crossing over p_C as well as the probability of mutation p_M were chosen depending on the size of the problem. For instance, in the case when $m = n = 20$, these settings were: $S = 500$, $T = 10000$, $p_C = 0.85$ and $p_M = 0.15$. In all experiments, fuzzy numbers were discretized at three levels of confidence: $\alpha_1 = 0$, $\alpha_2 = 0.5$ and $\alpha_3 = 1$.

For each pair (\tilde{A}, \tilde{b}) , the simulation was repeated 10 times, i.e. with 10 different initial populations. The solution with the smallest value of the cost function (9) was chosen as the final solution for the given system. Each solution was then compared with its corresponding exact solution and the Mean Square Errors (MSE) were computed. The experimental results for $m = n = 5$ and $m = n = 20$, in terms of the MSE are shown in Table 1:

Table 1. Test Results for $m = n = 5$ and $m = n = 20$

| | 5 × 5 systems | 20 × 20 systems |
|---------|---------------|-----------------|
| Min MSE | 7.17e-4 | 5.33e-2 |
| Max MSE | 1.72e-1 | 1.05e-1 |
| Avg MSE | 3.94e-2 | 6.66e-2 |

6 Conclusions

In this paper, a GA-based approach for approximating the classical solution of systems of linear fuzzy equations was proposed. As finding the classical solution of (5) is a complex task, the original SLFE was viewed in terms of an optimization problem. The cost function was built based on the Hausdorff metric on the family of fuzzy real numbers. Next a steady state strategy was designed to minimize this cost function, involving fuzzy arithmetic crossover and nonuniform mutation.

One of the issues which should be emphasized is that the given approach provides a possibility to solve large scale systems of linear fuzzy equations, which is the case in many real world applications. It can also be observed that when there is no exact extension principle based solution for a given SLFE, a *possible solution* can still be found. The technique discussed here is not limited to the case of square systems, but it is also capable of solving systems of m equations with n unknowns, where $m \neq n$.

The only type of uncertainty mentioned in the systems presented so far is impreciseness. In general, impreciseness can be modeled by interval numbers or fuzzy numbers - when additional knowledge is available. However in many practical problems, beside impreciseness, one may have to deal with another kind of uncertainty which is randomness [2]. Impreciseness is connected with values, concepts, rules... which can not be defined exactly, whereas randomness is related to events which may or may not take place in the future. Hence in many cases, the systems of linear equations under concern may contain both imprecise and random factors. This will be subject for further study by the authors.

References

1. Alefeld G., Herzberger J., *Introduction to Interval Computation*, Academic Press, London, 1983.
2. Bertoluzza C., Gil M.A., Ralescu D.A. (eds.), *Statistical Modeling, Analysis and Management of Fuzzy Data*, Physical-Verlag, New York, 2002.
3. Buckley J.J., Qu Y., Solving Systems of Linear Fuzzy Equations, *Fuzzy Sets and Systems*, **43**, 1991, 33-43.
4. Friedman M., Ma M., Kandel A., Fuzzy Linear Systems, *Fuzzy Sets and Systems*, 1998, 201-209.
5. Fuller R., Giove S., A neuro-fuzzy approach to FMOLP problems, *Proceedings of CIFT'94*, 1994, 97-101.
6. Goldberg, D. E.: *Genetic Algorithms in Search, Optimization and Machine Learning*. Kluwer Academic Publishers, Boston, MA, 1989.
7. Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, Berlin, 1996.
8. Neumaier A., *Interval Methods for Systems of Linear Equations*, Cambridge University Press, 1990.
9. Wang X., Zhong Z., Ha M., Iteration Algorithms for Solving a System of Fuzzy Linear Equations, *Fuzzy Sets and Systems*, **119**, 2001, 121-128.
10. Viet N.H., Kleiber M., AI Methods in Solving Systems of Interval Linear Equations, *Lecture Notes in Artificial Intelligence - ICAISC 2006*, **4029**, 2006, 150-159.
11. Zadeh L.A., Fuzzy Sets, *Information and Control*, **8**, 1965, 338-353.
12. Zadeh L.A., The Concept of a Linguistic Variable and its Application to Approximate Reasoning (I-III), *Information Science*, **8**, 1975, 199-250, **9**, 43-80.

On Fuzzy Driven Support for SD-Efficient Portfolio Selection^{*}

Włodzimierz Ogryczak¹ and Andrzej Romaszek²

¹ Warsaw University of Technology, Institute of Control & Computation Engineering,
Warsaw, Poland

wogrycza@ia.pw.edu.pl

² Warsaw School of Economics, Collegium of Economic Analysis, Warsaw, Poland
romaszki@bluewin.ch

Abstract. The stochastic dominance (SD) is based on an axiomatic model of risk-averse preferences and therefore, the SD-efficiency is an important property of selected portfolios. As defined with a continuum of criteria representing some measures of failure in achieving several targets, the SD does not provide us with a simple computational recipe. While limiting to a few selected target values one gets a typical multiple criteria optimization model approximating the corresponding SD approach. Although, it is rather difficult to justify a selection of a few target values, this difficulty can be overcome with the effective use of fuzzy target values. While focusing on the first degree SD and extending the target membership functions to some monotonic utility functions we get the multiple criteria model which preserves the consistency with both the first degree and the second degree SD. Further applying the reference point methodology to the multiple criteria model and taking advantages of fuzzy chance specifications we get the method that allows to model interactively the preferences by fuzzy specification of the desired distribution. The model itself guarantees that every generated solution is efficient according to the SD rules.

1 Introduction

The portfolio optimization problem considered in this paper follows the classical formulation and is based on a single period model of investment. At the beginning of a period, an investor allocates his capital among various securities, thus assigning a nonnegative weight (share of the capital) to each security. During the investment period, a security generates a random rate of return. This results in a change of the capital invested (observed at the end of the period) which is measured by the weighted average of the individual rates of return.

Let $J = \{1, 2, \dots, n\}$ denote a set of securities considered for an investment. For each security $j \in J$, its rate of return is represented by a random variable R_j with a given mean $z_j = \mathbb{E}\{R_j\}$. Further, let $\mathbf{x} = (x_j)_{j=1,2,\dots,n}$ denote a vector of

^{*} The research was partially supported by the Ministry of Science and Information Society Technologies under grant 3T11C 005 27.

decision variables x_j expressing the weights defining a portfolio. To represent a portfolio, the weights must satisfy a set of constraints that form a feasible set \mathcal{P} . The simplest way of defining a feasible set is by a requirement that the weights must sum to one and short sales are not allowed, i.e. $\sum_{j=1}^n x_j = 1$ and $x_j \geq 0$ for $j = 1, \dots, n$. Hereafter, it is assumed that \mathcal{P} is a general LP feasible set given in a canonical form as a system of linear equations with nonnegative variables. Each portfolio \mathbf{x} defines a corresponding random variable $R(\mathbf{x}) = \sum_{j=1}^n R_j x_j$ that represents the portfolio rate of return. The mean rate of return for portfolio \mathbf{x} is given as $z(\mathbf{x}) = \mathbb{E}\{R(\mathbf{x})\} = \sum_{j=1}^n z_j x_j$. Following the seminal work by Markowitz [6], the portfolio optimization problem is modeled as a mean-risk bicriteria optimization problem where $z(\mathbf{x})$ is maximized and some risk measure $\varrho(\mathbf{x})$ is minimized. In the original Markowitz model [6] the risk is measured by the standard deviation or variance while several other risk measures have been later considered thus creating the entire family of mean-risk (Markowitz-type) models [2][5].

The Markowitz model is frequently criticized as not consistent with axiomatic models of preferences for choice under risk [13]. Models consistent with the preference axioms are based on the relations of stochastic dominance or on expected utility theory [3][14]. In stochastic dominance, uncertain returns (random variables) are compared by pointwise comparison of some performance functions constructed from their distribution functions. The right-continuous cumulative distribution function (cdf): $F_{R(\mathbf{x})}(\eta) = \mathbb{P}\{R(\mathbf{x}) \leq \eta\}$, is used to define the *first degree stochastic dominance* (FSD). The second function is derived from the cdf as $F_{R(\mathbf{x})}^{(2)}(\eta) = \int_{-\infty}^{\eta} F_{R(\mathbf{x})}(\xi) d\xi$ and it defines the *second degree stochastic dominance* (SSD). Function $F_{R(\mathbf{x})}^{(2)}$, used to define the SSD relation, can also be presented [10] as $F_{R(\mathbf{x})}^{(2)}(\eta) = \mathbb{E}\{(\eta - R(\mathbf{x}))_+\}$ where $(\cdot)_+$ denotes the nonnegative part. Hence, while function $F_{R(\mathbf{x})}$ expresses the probability of a shortfall to target η , function $F_{R(\mathbf{x})}^{(2)}$ measures the mean shortfall to the target. The weak relations of stochastic dominance (FSD or SSD) are defined by pointwise inequalities for all real targets: $R(\mathbf{x}') \succeq_{FSD} R(\mathbf{x}'')$ if $F_{R(\mathbf{x}')}(\eta) \leq F_{R(\mathbf{x}'')}(\eta)$ for all η , and respectively, $R(\mathbf{x}') \succeq_{SSD} R(\mathbf{x}'')$ if $F_{R(\mathbf{x}')}^{(2)}(\eta) \leq F_{R(\mathbf{x}'')}^{(2)}(\eta)$ for all η . We say that portfolio \mathbf{x}' *dominates* \mathbf{x}'' *under the FSD (SSD)* if $F_{R(\mathbf{x}')}^{(1)}(\eta) \leq F_{R(\mathbf{x}'')}^{(1)}(\eta)$ ($F_{R(\mathbf{x}')}^{(2)}(\eta) \leq F_{R(\mathbf{x}'')}^{(2)}(\eta)$, respectively) for all η , with at least one strict inequality. A feasible portfolio $\mathbf{x}^0 \in \mathcal{P}$ is called *FSD (SSD) efficient* if there is no $\mathbf{x} \in \mathcal{P}$ such that $R(\mathbf{x}) \succ_{FSD} R(\mathbf{x}^0)$ ($R(\mathbf{x}) \succ_{SSD} R(\mathbf{x}^0)$).

We consider T scenarios with probabilities p_t (where $t = 1, \dots, T$). We assume that for each random variable R_j its realization r_{jt} under the scenario t is known. Typically, the realizations are derived from historical data treating T historical periods as equally probable scenarios ($p_t = 1/T$). The realizations of the portfolio return $R(\mathbf{x})$ are given as $r_t(\mathbf{x}) = \sum_{j=1}^n r_{jt} x_j$ and the expected value can be computed as $z(\mathbf{x}) = \sum_{t=1}^T r_t(\mathbf{x}) p_t = \sum_{t=1}^T \left[\sum_{j=1}^n r_{jt} x_j \right] p_t$. Similarly, values of functions $F_{R(\mathbf{x})}^{(1)}(\eta)$ and $F_{R(\mathbf{x})}^{(2)}(\eta)$ can easily be computed then for any given

target η . Nevertheless, as defined with a continuum of criteria representing some measures of failure in achieving several targets, the stochastic dominance models do not provide us with a simple computational recipe. While limiting to a few selected target values one gets typical multiple criteria optimization models approximating the corresponding stochastic dominance approaches [7,8,9]. However, in practice, it is rather difficult to justify a selection of a few target values. This difficulty can be overcome with the effective use of fuzzy target values.

2 Fuzzy Targets

Let us focus on the FSD model. Although the right-continuous cdf defines the FSD relation, the left-continuous cdf $F_{R(\mathbf{x})}^{(1)}(\eta) = \mathbb{P}\{R(\mathbf{x}) < \eta\}$ can equivalently be used for this purpose. It provides a lucid interpretation of the FSD relation as pointwise comparison of downside (below-target) risk measures representing probabilities of not achieving given levels (targets), as well as it allows us to build a simple optimization model for those measures. Suppose one has preselected m return values $\eta_1 > \eta_2 > \dots > \eta_m$ as targets to evaluate probabilities of the corresponding shortfalls. Introducing m corresponding criteria

$$s_k(\mathbf{x}) = F_{R(\mathbf{x})}^{(1)}(\eta_k) \quad \text{for } k = 1, 2, \dots, m \tag{1}$$

one gets the multiple criteria portfolio optimization model:

$$\min\{(s_1(\mathbf{x}), s_2(\mathbf{x}), \dots, s_m(\mathbf{x})) : \mathbf{x} \in \mathcal{P}\} \tag{2}$$

Due to the use of the left-continuous cdf as criteria $s_k(\mathbf{x}) = F_{R(\mathbf{x})}^{(1)}(\eta_k)$, optimization (2) can be formulated as mixed integer linear programming problem

$$\begin{aligned} \min (y_1, y_2, \dots, y_m) \quad \text{s.t. } \mathbf{x} \in \mathcal{P}, \quad y_k = \sum_{t=1}^T b_{kt} p_t \quad \text{for } k = 1, \dots, m \\ Mb_{kt} \geq \eta_k - \sum_{j=1}^n r_{jt} x_j, \quad b_{kt} \in \{0, 1\} \quad \text{for } k = 1, \dots, m; t = 1, \dots, T \end{aligned}$$

where M a sufficiently large constant and b_{kt} are binary variables taking value 1 whenever realization of the portfolio return under scenario t is below the target value η_k (i.e., $b_{kt} \geq \text{sign}((\eta_k - r_t(\mathbf{x}))_+)$).

Model (2) represents an approximation to the FSD approach. One can easily notice that any portfolio $\bar{\mathbf{x}} \in \mathcal{P}$ efficient (Pareto-optimal) solution to (2) can be FSD dominated only by an alternative efficient portfolio $\mathbf{x} \in \mathcal{P}$ with the same values of criteria $s_k(\mathbf{x}) = s_k(\bar{\mathbf{x}})$ for all $k = 1, \dots, m$. Hence, a small number of targets results in serious threat of FSD ambiguity in the sense the selected portfolio efficient to the multiple criteria model can easily be FSD dominated by another quite a different portfolio with the same values of the cdf at the few targets.

Note that values η_k actually represent the targets defined as corresponding (closed) crisp sets $\{\eta \in \mathbb{R} \mid \eta \geq \eta_k\}$, and their characteristic functions

$$\chi_k(\eta) = \begin{cases} 0 & \text{for } \eta < \eta_k \\ 1 & \text{for } \eta \geq \eta_k \end{cases}$$

allow to express the FSD criteria as $s_k(\mathbf{x}) = \mathbb{E}\{1 - \chi_k(R(\mathbf{x}))\}$. Taking advantages of the discrete distribution we have assumed they can be written as $s_k(\mathbf{x}) = 1 - \sum_{t=1}^T \chi_k(r_t(\mathbf{x}))p_t$. We propose to replace crisp targets with fuzzy targets C_k . We will focus on trapezoidal fuzzy sets defined by nondecreasing piecewise linear membership functions:

$$\mu_k(\eta) = \begin{cases} 0 & \text{for } \eta < \eta_k^- \\ (\eta - \eta_k^-) / (\eta_k^+ - \eta_k^-) & \text{for } \eta_k^- \leq \eta < \eta_k^+ \\ 1 & \text{for } \eta \geq \eta_k^+ \end{cases} \tag{3}$$

defined by two parameters (breakpoints) η_k^- and η_k^+ representing the largest return with membership level 0 and the smallest return with membership level 1, respectively. To relate them with values η_k defining the crisp targets we will assume that $\eta_k^+ = \eta_k$ while $\eta_k^- = \eta_k - \Delta_k$ for a given fuzzification parameter $\Delta_k > 0$. Hence, the fuzzy targets C_k can be specified by the interval $[\eta_k^-, \eta_k^+]$ or equivalently by the pair of numbers (η_k, Δ_k) .

With fuzzy targets C_k , the corresponding FSD criteria are expressed as

$$s_k(\mathbf{x}) = \mathbb{E}\{1 - \mu_k(R(\mathbf{x}))\} = \frac{1}{\Delta_k} \int_{\eta_k - \Delta_k}^{\eta_k} F_{R(\mathbf{x})}(\eta) \, d\eta \tag{4}$$

Applying them in multiple criteria model (2) one gets the fuzzy portfolio optimization model which offers more intuitive way to define targets. It provides an opportunity to define expectations in fuzzy terms like: “minimize the probability of shortfall to *medium* profit.” Moreover, due to aggregation on intervals $[\eta_k - \Delta_k, \eta_k]$ made in (4), the following assertion is valid.

Proposition 1. *Except for portfolios with identical cdf values within all intervals $[\eta_k - \Delta_k, \eta_k]$, every portfolio $\mathbf{x} \in \mathcal{P}$ efficient to the multiple criteria problem (2) with fuzzy defined criteria (4) is FSD efficient.*

Proof. Let $\bar{\mathbf{x}} \in \mathcal{P}$ be an efficient solution to the multiple criteria problem (2) with criteria (4). IF it is FSD dominated by portfolio $\mathbf{x} \in \mathcal{P}$, then $F_{R(\mathbf{x})}(\eta) \leq F_{R(\bar{\mathbf{x}})}(\eta)$ for all η with at least one inequality strict. Hence, due to (4), $s_k(\bar{\mathbf{x}}) \geq s_k(\mathbf{x})$ for all k and any strict inequality $F_{R(\mathbf{x})}(\eta) < F_{R(\bar{\mathbf{x}})}(\eta)$ for some $\eta \in [\eta_k - \Delta_k, \eta_k]$ would result in strict inequality $s_k(\bar{\mathbf{x}}) > s_k(\mathbf{x})$ thus contradicting the efficiency of $\bar{\mathbf{x}}$ to (2).

Note that distributions of rates of return are usually characterized by bounded support which can easily be covered by the interval $[\eta_m - \Delta_m, \eta_1]$ while all $\Delta_k \geq \eta_k - \eta_{k+1}$. Following Proposition 1, every portfolio efficient to the corresponding multiple criteria problem (2) with fuzzy defined criteria (4) is then (unconditionally) FSD efficient.

Problem (2) with fuzzy defined criteria (4) can be expressed as

$$\min\{(1 - \mathbb{E}\{\mu_1(R(\mathbf{x}))\}), 1 - \mathbb{E}\{\mu_2(R(\mathbf{x}))\}, \dots, 1 - \mathbb{E}\{\mu_m(R(\mathbf{x}))\}\} : \mathbf{x} \in \mathcal{P}\} \quad (5)$$

Hence, taking advantages of the discrete distributions, it can be formulated as the following mixed integer programming problem:

$$\begin{aligned} \min (y_1, y_2, \dots, y_m) \quad \text{s.t. } \mathbf{x} \in \mathcal{P}, \quad y_k &= \sum_{t=1}^T v_{kt} p_t \quad \text{for } k = 1, \dots, m \\ v_{kt} &\geq b_{kt}, \quad b_{kt} \in \{0, 1\} \quad \text{for } k = 1, \dots, m; t = 1, \dots, T \\ \Delta_k v_{kt} + M b_{kt} &\geq \eta_k - \sum_{j=1}^n r_{jt} x_j \quad \text{for } k = 1, \dots, m; t = 1, \dots, T \end{aligned} \quad (6)$$

where M a sufficiently large constant and b_{kt} are binary variables taking value 1 whenever realization of the portfolio return under scenario t is below the value $\eta_k - \Delta_k$ while $v_{kt} \geq 1 - \mu_k(r_t(\mathbf{x}))$.

3 Extension to Concave Utility

Membership functions μ_k in criteria $1 - \mathbb{E}\{\mu_k(R(\mathbf{x}))\}$ of problem (5) may be interpreted as utility functions used to maximize the corresponding expected utility value. In order to guarantee full consistency of the expected utility maximization with FSD the utility function must be strictly increasing. Moreover concave utility function represent risk aversion (SSD consistency). Membership function μ_k is strictly increasing within the interval $[\eta_k^-, \eta_k^+]$ while being constant for η smaller than η_k^- or larger than η_k^+ . Moreover, it is neither concave nor convex. Nevertheless, one can easily extend μ_k to a strictly increasing concave utility function $\bar{\mu}_k$ preserving the original values on interval $[\eta_k^-, \eta_k^+]$. The simplest such an extension is given by the following piecewise linear function:

$$\bar{\mu}_k(\eta) = \begin{cases} (\eta - \eta_k^-)/(\eta_k^+ - \eta_k^-) & \text{for } \eta < \eta_k^+ \\ 1 + \beta(\eta - \eta_k^+)/(\eta_k^+ - \eta_k^-) & \text{for } \eta \geq \eta_k^+ \end{cases} \quad (7)$$

where $0 < \beta < 1$ is an arbitrary small positive constant (Fig. 1). Note that function (7) can alternatively be expressed in terms of $\eta_k^+ = \eta_k$ and $\eta_k^- = \eta_k - \Delta_k$ as

$$\bar{\mu}_k(\eta) = 1 - \beta(\eta_k - \eta)/\Delta_k - (1 - \beta)(\eta_k - \eta)_+/\Delta_k \quad (8)$$

Reformulating (5) with functions $\bar{\mu}_k$ we get the multiple criteria problem

$$\min\{(1 - \mathbb{E}\{\bar{\mu}_1(R(\mathbf{x}))\}), 1 - \mathbb{E}\{\bar{\mu}_2(R(\mathbf{x}))\}, \dots, 1 - \mathbb{E}\{\bar{\mu}_m(R(\mathbf{x}))\}\} : \mathbf{x} \in \mathcal{P}\} \quad (9)$$

Since $\bar{\mu}_k$ are no longer fuzzy membership functions, the problem is not a fuzzy optimization model. It is a fuzzy driven multiple criteria model as the utility functions are defined by parameters of fuzzy targets. However, the use of utility

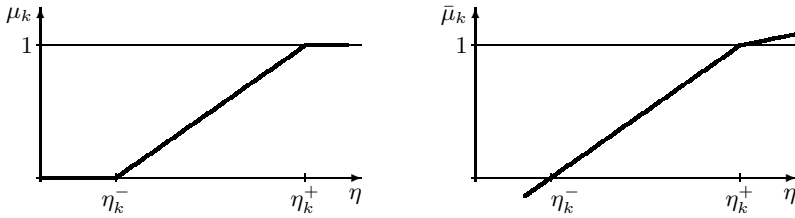


Fig. 1. Membership functions μ_k and increasing concave utility function $\bar{\mu}_k$

functions in (9) guarantees the model consistency with the SD rules. Note that, due to (8)

$$\begin{aligned}
 1 - \mathbb{E}\{\bar{\mu}_k(R(\mathbf{x}))\} &= \frac{\beta}{\Delta_k}(\eta_k - \mathbb{E}\{R(\mathbf{x})\}) + \frac{(1 - \beta)}{\Delta_k} \int_{-\infty}^{\eta_k} F_{R(\mathbf{x})}(\eta) \, d\eta \\
 &= [\beta(\eta_k - z(\mathbf{x})) + (1 - \beta)F_{R(\mathbf{x})}^{(2)}(\eta_k)]/\Delta_k
 \end{aligned}
 \tag{10}$$

thus allowing us to prove the following consistency statements.

Proposition 2. Every portfolio $\mathbf{x} \in \mathcal{P}$ efficient to multiple criteria problem (9) is FSD efficient.

Proof. Suppose that $\bar{\mathbf{x}} \in \mathcal{P}$ efficient to (9) is FSD dominated by portfolio $\mathbf{x} \in \mathcal{P}$, i.e. $F_{R(\mathbf{x})}(\eta) \leq F_{R(\bar{\mathbf{x}})}(\eta)$ for all η with at least one inequality strict. Then $z(\mathbf{x}) > z(\bar{\mathbf{x}})$ and $F_{R(\mathbf{x})}^{(2)}(\eta_k) \leq F_{R(\bar{\mathbf{x}})}^{(2)}(\eta_k)$ for all k . Hence, due to (10), $1 - \mathbb{E}\{\bar{\mu}_k(R(\bar{\mathbf{x}}))\} > 1 - \mathbb{E}\{\bar{\mu}_k(R(\mathbf{x}))\}$ for all k thus contradicting the efficiency of $\bar{\mathbf{x}}$ to (9).

Proposition 3. Except for portfolios with identical values of the expected utility criteria $\mathbb{E}\{\bar{\mu}_k(R(\mathbf{x}))\}$, every portfolio $\mathbf{x} \in \mathcal{P}$ efficient to the multiple criteria problem (9) is SSD efficient.

Proof. If $\bar{\mathbf{x}} \in \mathcal{P}$ is SSD dominated by $\mathbf{x} \in \mathcal{P}$, then $F_{R(\mathbf{x})}^{(2)}(\eta) \leq F_{R(\bar{\mathbf{x}})}^{(2)}(\eta)$ for all η and $z(\mathbf{x}) \geq z(\bar{\mathbf{x}})$ (10). Hence, due to (10), $1 - \mathbb{E}\{\bar{\mu}_k(R(\bar{\mathbf{x}}))\} > 1 - \mathbb{E}\{\bar{\mu}_k(R(\mathbf{x}))\}$ for all k and actually $1 - \mathbb{E}\{\bar{\mu}_k(R(\bar{\mathbf{x}}))\} = 1 - \mathbb{E}\{\bar{\mu}_k(R(\mathbf{x}))\}$ for all k due to the efficiency of $\bar{\mathbf{x}}$ to (9).

Taking advantages of the discrete distributions, problem (9) can be formulated as the following linear programming problem. Note that concave piecewise linear utility functions replacing the fuzzy membership functions allow us to eliminate binary variables from the optimization problem thus leading to the following linear programming model

$$\begin{aligned}
 \min (y_1, y_2, \dots, y_m) \quad \text{s.t. } \mathbf{x} \in \mathcal{P}, \quad y_k &= \sum_{t=1}^T v_{kt} p_t \quad \text{for } k = 1, \dots, m \\
 \Delta_k v_{kt} &\geq \beta(\eta_k - \sum_{j=1}^n r_{jt} x_j) \quad \text{for } k = 1, \dots, m; \quad t = 1, \dots, T \\
 \Delta_k v_{kt} &\geq \eta_k - \sum_{j=1}^n r_{jt} x_j \quad \text{for } k = 1, \dots, m; \quad t = 1, \dots, T
 \end{aligned}
 \tag{11}$$

4 Reference Point Method and Fuzzy Chances

An operational use of multiple criteria model (9) requires to select one efficient portfolio for implementation. This can be achieved with the so-called quasisatisficing approach to multiple criteria optimization introduced as the reference point method [15] and later extended the aspiration/reservation based decision support (ARBDS) approach with many successful applications [4]. In the ARBDS interactive scheme the decision maker (DM) specifies requirements in terms of aspiration and reservation levels, i.e., by introducing acceptable and required values for several criteria. Depending on the specified aspiration and reservation levels, a special scalarizing achievement function is built which generates an efficient solution to the multiple criteria problem when maximized. The computed efficient solution is presented to the DM as the current solution in a form that allows comparison with the previous ones and modification of the aspiration and reservation levels if necessary.

The scalarizing achievement function must be strictly monotonic with respect to each outcome (decreasing for the minimization problem (9)). Second, a solution with all individual outcomes y_k satisfying the corresponding reservation levels is preferred to any solution with at least one individual outcome worse than its reservation level. Next, provided that all the reservation levels are satisfied, a solution with all individual outcomes y_k equal to the corresponding aspiration levels is preferred to any solution with at least one outcome worse than its aspiration level. The generic scalarizing achievement function takes the following form [15]:

$$a(\mathbf{y}) = \min_{1 \leq k \leq m} \{a_k(y_k)\} + \varepsilon \sum_{i=k}^m a_k(y_k) \tag{12}$$

where ε is an arbitrary small positive number and a_k are the partial achievement functions measuring actual achievement of outcome y_k with respect to the corresponding aspiration and reservation levels ($y_k^a < y_k^r$, respectively). Thus the scalarizing achievement function is, essentially, defined by the worst partial achievement but additionally regularized with the sum of all partial achievements. The regularization term is introduced only to guarantee the solution efficiency in the case when the maximization of the main term (the worst partial achievement) results in a non-unique optimal solution.

The partial achievement function a_k can be interpreted as a measure of the DM's satisfaction with the current value of outcome of the k th criterion. It is a strictly decreasing function of outcome y_k with value $a_k = 1$ if $y_k = y_k^a$, and $a_k = 0$ for $y_i = y_k^r$. Various functions can be built meeting those requirements. We use the piecewise linear partial achievement function given by

$$a_k(y_k) = \begin{cases} (y_k^r - y_k)/(y_k^r - y_k^a) & \text{for } y_k > y_k^a \\ 1 + \alpha(y_k^a - y_k)/(y_k^r - y_k^a) & \text{for } y_k \leq y_k^a \end{cases} \tag{13}$$

where α is arbitrarily defined small parameter satisfying $0 < \alpha < 1$ representing additional increase of the DM's satisfaction over level 1 when a criterion

generates outcomes better than the corresponding aspiration level. Partial achievement function (13) is strictly decreasing and concave piecewise linear function, which guarantees its LP computability with respect to outcomes y_k . Finally, maximization of the entire scalarizing achievement function (12) can be implemented by auxiliary LP constraints thus preserving the LP structure of the entire RPM model:

$$\max\{a(\mathbf{y}) : \mathbf{x} \in \mathcal{P}, \quad y_k = 1 - \mathbb{E}\{\bar{\mu}_k(R(\mathbf{x}))\} \quad \text{for } k = 1, \dots, m\} \quad (14)$$

Model (14) generates efficient solutions to the corresponding problem (9). Hence, due to Propositions 2 and 3, the following assertions are valid.

Proposition 4. *Every portfolio $\mathbf{x} \in \mathcal{P}$ optimal to the RPM problem (14) is FSD efficient.*

Proposition 5. *Except for portfolios with identical values of the expected utility criteria $\mathbb{E}\{\bar{\mu}_k(R(\mathbf{x}))\}$, every portfolio $\mathbf{x} \in \mathcal{P}$ optimal to the RPM problem (14) is SSD efficient.*

For outcomes between the reservation and the aspiration levels, the partial achievement function a_k can be interpreted as a trapezoidal membership function μ_k for a fuzzy generalization of the crisp target $\{y_k : y_k \leq y_k^a\}$. Hence, the partial achievement function (13), similar to (7), can be viewed as an extension of the fuzzy membership function to a strictly monotonic and concave utility. One may also notice that the aggregation scheme used to build the scalarizing achievement function (12) from the partial ones may be interpreted as some fuzzy aggregation operator using the ordered weighted averaging [16]. In other words, maximization of the scalarizing achievement function (12) is consistent with the fuzzy optimization methodology and the aspiration and reservation levels may be regarded as the specification of fuzzy targets for the criteria values. Actually, as the original criteria s_k represent the shortfall probabilities, the aspiration and reservation levels represent the fuzzy chances. For instance, to seek low chance of losses and high chance of medium profit one may specify two (criteria) fuzzy targets: $(\eta_1^-, \eta_1^+) = (0.00, 0.02)$ with $y_1^a = 0.0$ and $y_1^r = 0.02$ and $(\eta_2^-, \eta_2^+) = (0.02, 0.05)$ with $y_2^a = 0.2$ and $y_2^r = 0.5$. Sample list of such simple fuzzy terms for portfolio selection is given in Table 1.

In our initial tests we have applied the model to the portfolio selection within the set of assets representing 32 major stock exchange indexes from various countries.¹ The analysis was based on the period 04.07.1997 – 24.09.2004 with the quotations every 4 weeks. The tests [12] has confirmed easiness of the interactive preference specification with fuzzy targets and fuzzy chances as well as good quality of generated portfolios with respect to the classical Markowitz criteria. Here we present an illustrative example of a simple analysis performed using only simplified fuzzy terms from Table 1.

We start with the requirement to find a portfolio with *low chance of losses*. This requirement leads us to the single-criterion model ($m = 1$) defined by

¹ The quotation data supplied by <http://finance.yahoo.com>

Table 1. Sample list of fuzzy terms

| Fuzzy targets | Definition | (η^-, η^+) |
|---------------|--------------------------|--------------------|
| losses | rate of return below 2% | (0.00,0.02) |
| medium profit | rate of return about 5% | (0.02,0.05) |
| large profit | rate of return about 10% | (0.05,0.10) |
| Fuzzy chances | Definition | (y^r, y^a) |
| low chance | prob. below 2% | (0.00, 0.02) |
| medium chance | prob. about 50% | (0.20, 0.50) |
| high chance | prob. about 80% | (0.50, 0.80) |

Table 2. Sample portfolios characteristics

| | P1 | P2 | P3 | P4 |
|-------------------------|--------|--------|--------|--------|
| Expected return rate | 1.011 | 1.015 | 1.02 | 1.013 |
| Variance | 0.0008 | 0.0015 | 0.0044 | 0.0011 |
| Mean absolute deviation | 0.0284 | 0.029 | 0.0472 | 0.0254 |

the fuzzy target $(\eta^-, \eta^+) = (0.00, 0.02)$ and the aspiration/reservation pair $(y^r, y^a) = (0.00, 0.02)$. Note that, due to the use of model (14) with strictly monotonic utility function $\bar{\mu}$ all probabilities of underachievements are minimized despite only losses are specified as a fuzzy target. This results in a FSD efficient portfolio (Proposition 4) presented as portfolio P1 in Table 2 where simplified portfolio characteristics are reported. Next, we try to examine a possibility of achieving portfolios with higher returns by maximizing the chance of getting *medium* or *high* return, respectively. This leads us to the single-criterion models ($m = 1$) defined by the fuzzy targets $(\eta^-, \eta^+) = (0.02, 0.05)$ and $(\eta^-, \eta^+) = (0.05, 0.10)$, respectively. In both models the aspiration/reservation pair $(y^r, y^a) = (0.50, 0.80)$ is used to represent *high chance* of reaching the results. The corresponding portfolios are presented as P2 and P3 in Table 2. Again, following Proposition 4, both the portfolios are FSD efficient. When comparing to P1, portfolio P2 (high chance of medium return) is characterized by more than 25% increase of the expected return while preserving very similar level risk measured with mean absolute deviation (mean absolute deviation is a risk measure consistent with the FSD and SSD orders whereas variance not [8]). Portfolio P3 (high chance of high return) reaches almost doubled expected return but with similar increase of the mean absolute deviation. Finally, in order to find a portfolio with *low chance* of losses and *high chance* to achieve *medium* return we solve a bicriteria model ($m = 2$) defined by the fuzzy targets $(\eta_1^-, \eta_1^+) = (0.00, 0.02)$ and $(\eta_2^-, \eta_2^+) = (0.02, 0.05)$. Generated FSD efficient portfolio P4 is indeed characterized by medium return and low risk (mean absolute deviation). Thus, it might be accepted as the final solution.

5 Concluding Remarks

Application of the FSD criteria to a few fuzzy targets of rate of return with membership functions expanded to monotonic concave utility functions allows us to specify easily an FSD and SSD consistent multiple criteria portfolio optimization model. Further, formally FSD criteria allow us to define aspiration and reservation levels via fuzzy definition of chances. The resulting RPM model is SSD consistent while allowing for easily defined preference parameters controlling the interactive analysis. The complete model is LP computable in the case of discrete random variables (historical data). The initial tests [12] has confirmed easiness of the interactive preference specification with fuzzy targets and fuzzy chances as well as good quality of generated portfolios with respect to the classical Markowitz criteria.

References

1. Bawa, V.S.: Safety-first, stochastic dominance and optimal portfolio choice. *J. Finan. Quant. Anal.* **13** (1978) 252–271.
2. Krzemienowski, A., Ogryczak, W.: On extending the LP computable risk measures to account downside risk. *Comput. Optimization and Appls.*, **32**, (2005), 133–160.
3. Levy, H.: Stochastic dominance and expected utility: survey and analysis. *Manage. Sci.* **38** (1992) 555–593.
4. Lewandowski, A., Wierzbicki, A.P. (eds.): *Aspiration Based Decision Support Systems — Theory, Software and Applications*. Springer, Berlin, 1989.
5. Mansini, R., Ogryczak, W., Speranza, M.G.: On LP solvable models for portfolio selection. *Informatica*, **14**, (2003), 37–62.
6. Markowitz, H.M.: Portfolio selection. *J. Fin.* **7** (1952) 77–91.
7. McNamara, J.R.: Portfolio selection using stochastic dominance criteria. *Decision Sci.* **29** (1998) 785–801.
8. Ogryczak, W.: Multiple criteria linear programming model for portfolio selection. *Ann. Oper. Res.* **97** (2000) 143–162.
9. Ogryczak, W.: Multiple criteria optimization and decisions under risk. *Control & Cybernetics*, **31**, (2002), 975–1003.
10. Ogryczak, W., Ruszczyński, A.: From stochastic dominance to mean-risk models: semideviations as risk measures. *Eur. J. Oper. Res.* **116** (1999) 33–50.
11. Ribeiro, R.A., Zimmermann, H.-J., Yager, R.R., Kacprzyk, J. (eds.): *Soft Computing in Financial Engineering*, Physica-Verlag, Heidelberg, 1999.
12. Romaszkiwicz, A., Portfolio selection with techniques of fuzzy and multicriteria optimization (in Polish). Ph.D. Thesis, Warsaw School of Economics, 2006.
13. Rothschild, M., Stiglitz, J.E.: Increasing risk: I. A definition. *J. Econ. Theory* **2** (1969) 225–243.
14. Whitmore, G.A., Findlay, M.C. (eds.): *Stochastic Dominance: An Approach to Decision-Making Under Risk*, D.C.Heath, Lexington MA 1978.
15. Wierzbicki, A.P.: A mathematical basis for satisficing decision making. *Math. Modelling* **3** (1982) 391–405.
16. Yager, R.R., Filev, D.P.: *Essentials of Fuzzy Modelling and Control*, Wiley, NY, 1994.

Fuzzy Kernel Ridge Regression for Classification

YoungSik Choi and JiSung Noh

Department of Computer Engineering, Hankuk Aviation University,
200-1 Hwajun-Dong Dukyang-Gu
Goyang-City, Gyeonggi-Do, Korea
{choimail, jisung}@hau.ac.kr

Abstract. We present a robust version of kernel ridge regression for classification, which can gracefully handle outliers. We first show that the ridge regression can be reduced to the proximal support vector machine (PSVM) which has been successfully applied in classification problems. In order to incorporate robustness into kernel ridge regression, we reformulate and derive a fuzzy version of kernel ridge regression so that each sample can contribute to formation of a decision boundary according to its corresponding fuzzy class membership. We also present how to determine the fuzzy class membership values. Experiments over synthetic and real data sets demonstrate superiority of the proposed method, comparing with traditional methods such as support vector machines (SVMs).

1 Introduction

The ridge regression, also known as a least squares support vector machine (LS-SVM), can be reduced to the proximal support vector machine (PSVM) [2] when target values are set to ± 1 for positive and negative samples. As in the PSVM, the ridge regression for classification seeks two planes such that each plane is closet to one of two data sets to be classified and the two planes are as far apart as possible. The two planes are called proximal planes and the plane bisecting these two proximal planes is the decision boundary. The ridge regression can be generalized to a kernel ridge regression that can handle non-linear decision boundaries [7].

In this paper, we propose a robust version of kernel ridge regression (KRR) for classification. Since in many classification problems outliers are inevitable, sophisticated classifiers are required to have some capability of handling outliers so that the overall performance of classifiers should be less sensitive to outliers [4]. In order to incorporate the capability of handling outliers, i.e., robustness, into the kernel ridge regression, we reformulate the objective function of the KRR such that a sample contributes to the formation of a decision boundary to the extent of the degree of its class membership. We call this new kernel ridge regression a fuzzy version of KRR (Fuzzy KRR). As indicated in [5][8], the method of computation of fuzzy class membership largely influences the performance of fuzzy classifiers. One can predetermine the membership values for training samples according to the confidence level of class membership. However, in most applications, it may be difficult to assign the membership values to training samples in advance.

In this paper, we propose to determine the fuzzy membership values from a monotonic decreasing function of an error caused by a training sample \mathbf{x}_i . Since the amount of error from \mathbf{x}_i is proportional to its corresponding proximal planes in kernel ridge regression, we can interpret the error as a distance from \mathbf{x}_i to its corresponding proximal plane [9]. Therefore, samples far apart from two proximal planes would get very low values of fuzzy membership, and consequently would make very little contribution to formation of the decision boundary. The membership values for all training samples are initialized to ‘1’. After obtaining the decision boundary, we compute errors from the proximal planes, update the membership values according to the errors, and regenerate the decision boundary with new membership values. We do this process until there is no significant change in membership values. In this fashion, the Fuzzy KRR can produce more reliable decision boundary.

There are a few approaches to address the aforementioned problems in the literature [3][5][6]. However, they did not clearly provide how to determine the membership values although all these approaches addressed the problem to incorporate the fuzzy membership into the SVM processes. On the other hand, we provide an efficient training method such that the membership value of a sample should be inversely proportional to the error from that sample.

In Sect. 2, we briefly introduce the classic kernel ridge regression. In Sect. 3, we present a fuzzy version of kernel ridge regression and the training method. In Sect. 4, we present experimental results from the proposed method on synthetic and real data sets, comparing with other traditional methods including the SVM. We make conclusions of this work in Sect. 5.

2 Kernel Ridge Regression

Ridge regression is a method from classical statistics that implements a regularized form of least squares regression [7][12]. Suppose that we are given a training data set $\mathbf{S} = \{(\mathbf{x}_i, y_i)\}$, where \mathbf{x}_i is a data point in an n dimensional real space \mathfrak{R}^n and y_i is a real number. Ridge regression can be stated as a minimization problem with respect to \mathbf{w} and b in $f(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle - b = \mathbf{y}$

$$\text{minimize } \frac{C}{2} \|\boldsymbol{\xi}\|^2 + \frac{1}{2} \left\| \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix} \right\|^2, \quad \text{subject to: } \mathbf{X}\mathbf{w} - b\mathbf{e} + \boldsymbol{\xi} = \mathbf{y} \tag{1}$$

In equation (1), all vectors are column vectors, and the training samples are represented as a matrix $\mathbf{X} \in \mathfrak{R}^{m \times n}$, where m denotes a number of training samples and n represents a dimensionality of \mathbf{x}_i . Vector \mathbf{y} represents a column vector whose element represents y_i . Vector $\boldsymbol{\xi}$ denotes errors (or slack variables) and C is a non-zero constant, and \mathbf{e} represents a column vector of ‘1’. Introducing Lagrange multipliers $\boldsymbol{\alpha}$ into equation (1), one can solve the problem by minimizing the following objective function.

$$L(\mathbf{w}, b, \xi, \mathbf{a}) = \frac{C}{2} \|\xi\|^2 + \frac{1}{2} \left\| \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix} \right\|^2 - \mathbf{a}^T (\mathbf{X}\mathbf{w} - b\mathbf{e} + \xi - \mathbf{y}) \tag{2}$$

Taking the first derivative of equation (2) with respect to \mathbf{w} , b , ξ and setting it equal to zero, one can obtain the equations: $\mathbf{w} = \mathbf{X}^T \mathbf{a}$, $b = -\mathbf{e}^T \mathbf{a}$, $\xi = \mathbf{a}/C$, and $\mathbf{a} = (\mathbf{X}\mathbf{X}^T + \mathbf{e}\mathbf{e}^T + \mathbf{I}/C)^{-1} \mathbf{y}$. Hence, the corresponding linear model $f(\mathbf{x})$ becomes

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle - b = (\mathbf{x}^T \mathbf{X}^T + \mathbf{e}^T) \mathbf{a} \tag{3}$$

It is noteworthy to point out that the proximal support vector machine [2] (PSVM) is a special form of ridge regression where \mathbf{y} takes only values of ± 1 , representing positive and negative class memberships. In this case, the ridge regression seeks two parallel proximal planes such that one plane $\langle \mathbf{w} \cdot \mathbf{x} \rangle - b = +1$ is closet to positive samples and the other $\langle \mathbf{w} \cdot \mathbf{x} \rangle - b = -1$ is closet to negative samples while the two planes are as far apart as possible.

A kernel ridge regression is a non-linear version of ridge regression [7] that can be obtained via the so-called ‘‘kernel trick’’, whereby a linear ridge regression model is constructed in a higher dimensional feature space F , where $F = \{\phi(\mathbf{x}) \mid \mathbf{x} \in \mathbf{X}\}$. The inner product in feature space is computed via a kernel function $K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle$. The kernel function may be any positive definite ‘‘Mercer’’ kernel such as Gaussian radial basis function $K(\mathbf{x}, \mathbf{z}) = \exp(-k\|\mathbf{x} - \mathbf{z}\|^2)$. A kernel-induced ridge regression, simply kernel ridge regression, can be formulated as the following minimization problem in a feature space:

$$\text{minimize } \frac{C}{2} \|\xi\|^2 + \frac{1}{2} \left\| \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix} \right\|^2 \quad \text{subject to: } \phi(\mathbf{X})\mathbf{w} - b\mathbf{e} + \xi = \mathbf{y} . \tag{4}$$

One can solve equation (4) as in ridge regression. The corresponding non-linear model obtained becomes

$$f(\phi(\mathbf{x})) = \langle \mathbf{w} \cdot \phi(\mathbf{x}) \rangle - b = (K(\mathbf{x}^T, \mathbf{X}^T) + \mathbf{e}^T) \mathbf{a}, \tag{5}$$

where $\mathbf{a} = (K(\mathbf{X}, \mathbf{X}^T) + \mathbf{e}\mathbf{e}^T + \mathbf{I}/C)^{-1} \mathbf{y}$. In equation (5), we used the notation as described in [2]. For $\mathbf{A} \in \mathfrak{R}^{m \times n}$ and $\mathbf{B} \in \mathfrak{R}^{n \times k}$, the kernel $K(\mathbf{A}, \mathbf{B})$ maps $\mathfrak{R}^{m \times n} \times \mathfrak{R}^{n \times k}$ into $\mathfrak{R}^{m \times k}$. In particular, if \mathbf{x} and \mathbf{y} are column vectors in \mathfrak{R}^n then, $K(\mathbf{x}^T, \mathbf{y})$ is a real number, $K(\mathbf{x}^T, \mathbf{A}^T)$ is a row vector in \mathfrak{R}^m and $K(\mathbf{A}, \mathbf{A}^T)$ is an $m \times m$ matrix.

3 Fuzzy Kernel Ridge Regression for Classification

In this section, we present a fuzzy version of kernel ridge regression for classification and its corresponding training method. Note that as in the PSVM, the kernel ridge regression for classification seeks the decision boundary, and correspondingly two proximal planes.

3.1 Fuzzy Kernel Ridge Regression

In equation (4), the kernel ridge regression seeks a non-linear model of $f(\phi(\mathbf{x}))$, i.e., a non-linear decision boundary such that $\|\xi\|^2$, the total sum of squares errors should be minimized while two proximal planes are as far apart as possible. In our fuzzy version of KRR, we reformulate equation (4) such that the contribution of \mathbf{x}_i to the decision boundary should be proportional to its fuzzy membership value. That is, if the class membership of \mathbf{x}_i is lower, \mathbf{x}_i should be less contributed to the formation of the decision boundary, and vice versa. In order to achieve this, we scale the error from sample \mathbf{x}_i , i.e., $\xi_i = y_i - f(\phi(\mathbf{x}_i))$ by its fuzzy membership value. The corresponding fuzzy version of kernel ridge regression in a matrix form can be stated as equation (6), where \mathbf{D} is a diagonal matrix whose element \mathbf{D}_{ii} represents a fuzzy membership value, $0 < \mathbf{D}_{ii} \leq 1$, of training sample \mathbf{x}_i . In equation (6), the errors from the samples with lower fuzzy membership values are treated less significantly so that the samples make less contribution to the formation of decision boundary.

$$\text{minimize } \frac{C}{2} \|\mathbf{D}\xi\|^2 + \frac{1}{2} \left\| \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix} \right\|^2 \quad \text{subject to: } \phi(\mathbf{X})\mathbf{w} - b\mathbf{e} + \xi = \mathbf{y} \quad (6)$$

Introducing Lagrange multipliers α , taking the first derivative of equation (6), and setting it equal to zero, one can obtain the following equations:

$$\begin{aligned} \mathbf{w} &= \phi(\mathbf{X}^T)\alpha, & b &= -\mathbf{e}^T\alpha, & \xi &= ((\mathbf{D}^2)^{-1}\alpha)/C, \\ \alpha &= (K(\mathbf{X}, \mathbf{X}^T) + \mathbf{e}\mathbf{e}^T + ((\mathbf{D}^2)^{-1})/C)^{-1}\mathbf{y}. \end{aligned} \quad (7)$$

Using these equations, a fuzzy non-linear decision function $f(\phi(\mathbf{x}))$ is obtained as the following. Note that α_i for \mathbf{x}_i is proportional to the membership value of \mathbf{x}_i .

$$f(\phi(\mathbf{x})) = \langle \mathbf{w} \cdot \phi(\mathbf{x}) \rangle - b = (K(\mathbf{x}^T, \mathbf{X}^T) + \mathbf{e}^T)\alpha. \quad (8)$$

3.2 Training Method

The membership values in the proposed version of KRR may be assigned beforehand as mentioned in [5]. However, it is difficult to predetermine the membership value in most real applications. In this paper, we propose to compute fuzzy membership values from a monotonic decreasing function of an error ξ_i caused by a training sample \mathbf{x}_i , which can be interpreted as a distance from \mathbf{x}_i to its corresponding proximal plane. Note that the two proximal planes can be interpreted as the optimal planes that represent most of training samples. Therefore, a training sample significantly apart from its proximal plane should get a lower membership value and a sample closer to the proximal plane should get a higher value. One can use any monotonic decreasing function of error ξ_i . In this paper, we used a bell-shaped membership function as in equation (9). If an absolute value of error ξ_i is less than a threshold value T , the membership value is assigned 1. We set the threshold value T to the mean value of errors and the constant k is set to the variance of errors.

$$D_{ii} = \begin{cases} \text{if } \text{abs}(\xi_i) < T, & 1 \\ \text{otherwise,} & \exp\left(-k\|\xi_i - T\|^2\right) \end{cases} \quad (9)$$

We first initialize all membership values of training samples into 1. Then, we run the Fuzzy KRR, obtain the decision boundary, and compute errors from the proximal planes. Then, using equation (9), we compute the membership values for training samples, and obtain new decision boundary. We do this process until no significant change in membership values. The whole procedure is summarized in Fig. 1.

Algorithm for the Fuzzy KRR

1. Initialize all the membership values \mathbf{D}_{ii} into 1.
 2. Obtain \mathbf{w} and b for the decision boundary using equation (7).
 3. Compute errors from all \mathbf{x}_i using $\xi_i = y_i - f(\phi(\mathbf{x}_i))$ and equation (8).
 4. Compute membership values \mathbf{D}_{ii} for all \mathbf{x}_i using equation (9).
 5. Go to step 2 until no significant change in \mathbf{D} ,
-

Fig. 1. Training algorithm of the Fuzzy KRR

4 Experimental Results

In this section we present the experimental results from the proposed method on various data sets, comparing with the traditional learning methods, including kernel SVM (KSVM) and KRR [1][4][7][11][15]. In Sect. 4.1, we first present the experimental results on a synthetic data set to illustrate the characteristics of the Fuzzy KRR, comparing with the classical SVMs. Then, we present the performance of the Fuzzy KRR on Web page classifications. The Fuzzy KRR has converged quickly in 3 or 4 iterations throughout the experiments.

4.1 Synthetic Data

In order to illustrate the properties of the Fuzzy KRR comparing with traditional SVMs, we generated a synthetic data set as in Fig. 2.(a). Both positive and negative samples, 100 points for each class, were generated from two different Gaussian distributions, and randomly chosen 15 outliers were added to each class. Positive samples and negative samples are depicted as \times and \circ , respectively. Fig. 2.(b), (c), and (d) show the experimental results from the SVM, KRR, and Fuzzy KRR, respectively. In these figures, the solid lines represent the decision boundaries obtained from each learning method. One can see that the results from the SVM and the KRR are almost equivalent. Note that both methods tend to overfit into training samples. On the other hand, the Fuzzy KRR appears to be less sensitive to outliers, extracting a hyperplane bisecting two classes. This simple experiment indicates that the proposed fuzzy version of KRR tolerates outliers better than the traditional SVMs.

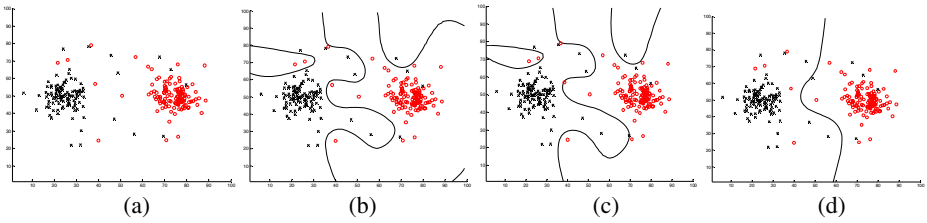


Fig. 2. Results on a synthetic data set where solid lines depict the decision boundaries: (a) training samples (b) result from the SVM (c) result from the KRR, and (d) result from the Fuzzy KRR

4.2 Web Page Classification

In order to evaluate the Fuzzy KRR method on real applications, we applied the Fuzzy KRR to Web page classification, comparing with the SVMs. Automatic categorization of Web pages is a critical step for mining the Web [10]. Various statistical learning algorithms have been applied to Web page classification [1][10-11][14-15]. Among a plethora of approaches, the support vector machines (SVM) have been well recognized for the outstanding performance in the literature [1][4][10-11].

For an experimental data set, we used the WebKB data set [13], well known for Web page classification problems. The Web KB data set was collected from the computer science departments from various universities, and has seven categories: course, department, faculty, project, staff, student, and others. In this paper, we used four categories including course, faculty, project, and student as shown in Table 1. For feature extraction, we removed stop words and also words with document frequency of less than 3, and words with document frequency of higher than 1000. After removing these words, we used a normalized term frequency vector as a feature vector. We used the leave-one-out cross validation method for the evaluation of the learning methods. For the performance measurement, we used the micro average F_1 measure as in [11][15], which is a harmonic average of recall and precision.

Table 1. WebKB data

| Category | Course | Faculty | Project | Student |
|-------------|--------|---------|---------|---------|
| Sample size | 930 | 1124 | 504 | 1641 |

The experiments have been done with varying the number of training samples and outliers, where the outliers were randomly added to each class. We only show two experimental results in Table 2 due to the space limit. Column (a) in Table 2 shows the results from 350 training samples for each class and 11 outliers. Column (b) shows the experimental results from 75 % of training samples for each class and 11 outliers. One can see that in general the Fuzzy KRR shows much better classification performance than KSVM and KRR when there were outliers presented.

Table 2. Experimental results from (a) 350 training samples and (b) 75% training samples for each class with 11 outliers

| Class | (a) F_1 measure (%) | | | (b) F_1 measure (%) | | |
|---------|-----------------------|-------|--------------|-----------------------|-------|--------------|
| | KSVM | KRR | Fuzzy KRR | KSVM | KRR | Fuzzy KRR |
| Course | 90.54 | 92.73 | 93.51 | 93.94 | 94.95 | 95.41 |
| Faculty | 76.93 | 78.38 | 79.10 | 82.73 | 83.24 | 83.62 |
| Project | 49.72 | 51.46 | 52.42 | 74.05 | 73.97 | 74.96 |
| Student | 84.34 | 86.13 | 86.99 | 89.52 | 89.33 | 89.70 |

5 Conclusion

We have presented a fuzzy version of kernel ridge regression in order to deal with the outliers in classification problems. In the proposed Fuzzy KRR, we determine the membership values according to the distances from the proximal planes such that samples, apart from the proximal planes, take much less influence on formation of the decision boundary. Experimental results on various data sets indicate that our approach performs better than the traditional SVMs. The proposed Fuzzy KRR can be applied for the classification problems with training samples collected from noisy environments.

Acknowledgements. “This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD)”(KRF-2004-041-D00606).

References

1. A. Basu, C. Watters, M. Shepherd: Support Vector Machines for Text Categorization. HICSS, Vol. 4 (2003)
2. Glenn Fung, Olvi L. Mangasaian: Proximal Support Vector Machine Classifiers. KDD ACM (2001) 64–70
3. Jayadeva, Reshma Khemchandani, Suresh Chandra: Fast and Robust Learning through Fuzzy Linear Proximal Support Vector Machines. Neurocomputing 61 (2004) 401–411
4. Richard O. Duda, Peter E. Hart, David G. Stork: Pattern Classification. Wiley-Interscience (2001)
5. Chun-Fu Lin, Sheng-De Wang: Fuzzy Support Vector Machines. Neural Networks, IEEE Transaction on, Vol. 13, Issue. 2 (2002) 464–471
6. Ling Jing: A Robust Proximal Support Vector Machine for Classification. Proc. International Conf. on ICNN&B, IEEE, Vol. 1 (2005) 576–580
7. Nello Cristianini, John Shawe-taylor: An Introduction to Support Vector Machine and other kernel-based learning methods. Cambridge University Press (2000)
8. George J. Klir, Tina A. Folger: Fuzzy Sets, Uncertainty, and Information. Prentice Hall (1988)
9. YoungSik Choi, JiSung Noh: Relevance Feedback for Content-Based Image Retrieval Using Proximal Support Vector Machine. LNCS 3044 (2004) 942–951

10. Soumen Chakrabarti: Mining the Web : Discovering Knowledge from Hypertext Data. Morgan Kaufmann (2003)
11. Thorsten Joachims: A statistical learning model of text classification for support vector machines. SIGIR ACM (2001) 128–136
12. C. Saunders, A. Gammerman, V. Vovk: Ridge Regression learning Algorithm in Dual Variables. In Proc. 15th Int. Conf. on Machine Learning, Madison, WI, (1998) 515–521.
13. <http://www.cs.cmu.edu/~webkb/>
14. Y. Yang: An Evaluation of Statistical Approaches to Text Categorization. Journal of Information Retrieval (1999) 67–88
15. Thorsten Joachims: Text Categorization with Support Vector Machines: Learning with Many Relevant Feature. In Proc. 10th Eur. Conf. on ML (1998) 137–142

Assessment of the Accuracy of the Process of Ceramics Grinding with the Use of Fuzzy Interference

Dariusz Lipiński and Wojciech Kacalak

Koszalin University of Technology, Faculty of Mechanical Engineering,
75-620 Koszalin, Poland

{dariusz.lipinski,wojciech.kacalak}@tu.koszalin.pl

Abstract. Grinding of ceramic materials is an expensive process considering the cost of abrasive tools and a relatively small machining efficiency. This paper presents methods to increase the use of the technological potential of automatic machining of ceramic materials with a concurrent control of the machining accuracy. A method of an assessment of the influence of process variables monitored on the accuracy of the sizes and shape of ceramic elements machined was presented. An application of fuzzy interference methods facilitated a creation of a universal algorithm to assess the accuracy of machining independent of the accepted parameters and machining conditions.

1 Introduction

Ceramic materials, owing to its characteristic properties: a high degree of hardness, substantial brittleness, small thermal conductivity and diamagnetic properties are materials which are hard to process. Guaranteeing products from technical ceramics with a high dimensional and shape accuracy usually requires an operation of precise grinding. It is estimated that as many as 85% [1] of ceramic parts is subject to grinding as finishing. Grinding is an expensive kind of processing of ceramic materials. In the production of ceramic materials, the costs of machining are assessed to constitute even up to 80% of the overall costs of the product. These costs depend of the selection of grinding parameters and the level of the use of the method's technological potential [2]. A reduction of the costs of grinding through the application of a greater efficiency of allowance removal is limited by an increasing probability of damaging the ceramics surface [3, 4], which leads to deteriorated utilization properties or even a disqualification of the product.

Due to a significant share of auxiliary times in the times of a single abrasive machining operation of small ceramics elements, which are the result of a significant labour consumption of the setting and fixing of elements, automatic machining is an economically justified machining method [5]. It guarantees an automation of feeding, setting and fixing of the elements machined. An increase of the machining output of ceramics requires optimization methods for machining parameters to be developed. A combination of the procedures of an optimization

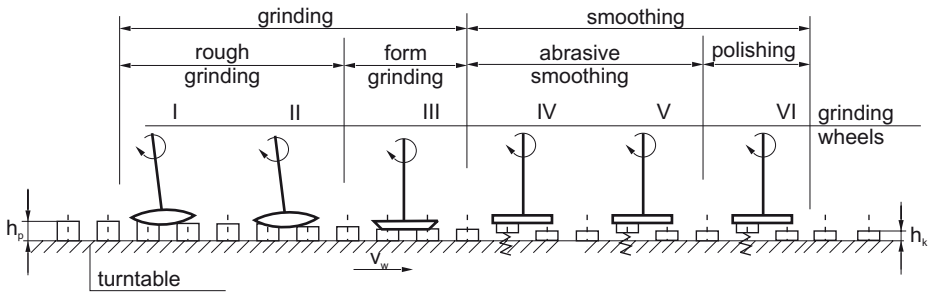


Fig. 1. A diagram of the method of an integrated machining of small ceramic elements in an automatic cycle

of machining parameters with the assessment methods of machining accuracy is a necessary condition for the achievement of the highest quality level of goods manufactures in expensive grinding methods.

This publication presents a system for evaluation a machining accuracy in the process of an automatic machining of ceramic materials with the use of fuzzy interference methods [6], [7]. In the machining method analysed, the total allowance to be removed was divided between N grinding wheels placed on the perimeter of the turntable (Fig. 1).

The result of the occurrence of inaccuracies in the process is a distortion of an optimal division of the allowance, and therefore a risk connected with the achievement of the required measurement and shape accuracy of the elements machined. A direct measurement of the machining parameters on individual fast headstocks during the machining process is difficult, an in many cases impossible to conduct. However, as these quantities depend of on the values of the process variables, such as the grinding force, power, the vibration of the machining system, and an acoustic emission, they can be assessed and forecast directly before the machining is completed [8].

2 Assumptions Concerning the Normalization of Process Variables

In the machining process analysed, a change of the value of the variable monitored is important only when it has an influence on the machining quality. A relative value of the variable monitored does not give any information on the level of its influence on the substantial parameters of the product's quality. In order to guarantee a universality of the system, normalized values were used. A normalization of the monitored variables was conducted with the use of the theory of fuzzy sets (Fig. 2). The normalized values of the process variable constitute diagnostic information (a diagnostic variable), which constitutes the basis for the assessment of the level of machining inaccuracy.

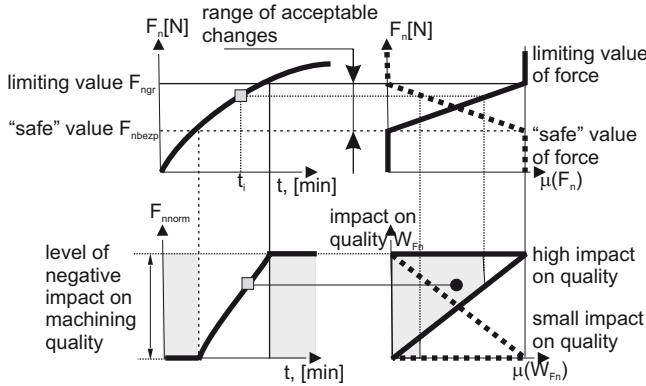


Fig. 2. Assumptions concerning the working principles of the fuzzy model for the assessment of the advancement level of the process' feature on the example of a componential of a normal grinding force

The range of the changes of the values of the diagnostic variable can be determined depending of its specificity and can be placed in the range $(-1, 1)$, $(0, 1)$, or $(0, -1)$. As a result of the representation, the process variable value becomes independent of the reference point (the initial value being the result of characteristic features of a given process). By a suitable setting of threshold values (determined as a result of optimization), identical values of the diagnostic variable can be obtained in different working conditions of the process. This also facilitates a correct classification of the states of the process monitored through making them independent of the absolute values of the process variables.

3 Normalization of Monitored Process Variables

The normalization of process variables requires a determination the limiting values of the monitored variables. The limiting values of the process variables were determined with the use of the relationship obtained as a result of experimental tests of the grinding process of ceramic elements. Experimental tests of the automated grinding process of ceramic elements [9] proved an influence of the machining parameters, i.e. the quantity of allowance on the i^{th} fast headstock a^{i} and the speed of the travelling of objects v_w on the normal component of grinding force F_n and the acceleration of the vibrations of the turntable \dot{v}_s . Also, dependencies were determined between the values of the process variables monitored (the normal component of grinding force F_n and the acceleration of the vibrations of the turntable \dot{v}_s) and the deviation of height Δh^{i} and flatness Δp^{i} of elements machined in the i^{th} machining zone.

In the optimization process of the machining parameters a complex objective function was a maximum efficiency of machining process Q_v , determined with the following dependence:

$$Q_v = \frac{\pi \cdot d^2}{4} \cdot \frac{v_w}{T_o} \cdot \sum_{i=1}^N a^{\{i\}}, \quad (1)$$

where: d - diameter of machined objects [mm], v_w - travelling speed of machined objects [mm/s], T_o - the circular pitch of the arrangement of objects on the grinder turntable [rad], N - number of fast headstocks, $a^{\{i\}}$ - amount of allowance for i^{th} fast headstock [mm].

In the optimization process, the influence of the machining process parameters (the allowance amount on i^{th} fast headstock $a^{\{i\}}$ and the travelling speed of objects v_w) on height deviation $\Delta h^{\{i\}}$ and flatness deviation $\Delta p^{\{i\}}$ of the elements machined in the i^{th} machining zone were taken into account:

$$\Delta h\{i\} = f_1^{\{i\}}(a^{\{i\}}, v_w), \Delta p\{i\} = f_2^{\{i\}}(a^{\{i\}}, v_w), \quad (2)$$

where: $f^{\{i\}}$ - the function which includes the influence of the parameters of the i^{th} fast headstock on the accuracy of the shape and size of the elements machined. The parameters of the function depend on the properties of the configuration of the machine tool-the object-the tool.

The limiting value of the deviations of the size and shape on the i^{th} fast headstock is the result of the technological capabilities of consecutive fast headstocks:

$$\Delta h_{max}^{\{i\}}(\Delta h_{max}^{\{i-1\}} + a^{\{i\}}, v_w) \leq \Delta h_{dop}^{\{i\}}, \Delta p_{max}^{\{i\}}(\Delta h_{max}^{\{i-1\}} + a^{\{i\}}, v_w) \leq \Delta p_{dop}^{\{i\}}, \quad (3)$$

where: $\Delta h_{dop}^{\{i\}}$, $\Delta p_{dop}^{\{i\}}$ - permissible amount of deviation of the size and shape on the i^{th} fast headstock, with is the result of the technological capabilities of further fast headstocks.

The acceptance of the limiting parameters of machining guarantees that the assumed parameters of the product's quality ($\Delta h_{dop}^{\{i\}} \leq 20\mu\text{m}$ and $\Delta p_{dop}^{\{i\}} \leq 3.5\mu\text{m}$ for the final form grinding stage) will be achieved on the final stage, however the occurrence in the process of any inaccuracies (e.g. abrasive and form wear occurring with time) will result in an increased share of defective products in the overall number of the machined elements.

The optimization process which leads to the determination of „safe” machining process parameters assumes the acceptance of an optimal division of allowance and searching of such a travelling speed of objects v_w for which an increase of the process variables monitored is possible (a component of the normal grinding force F_n and an acceleration of the vibrations of turntable \dot{v}_s) in the assumed permissible durability of the grinding wheel (Table I). This assumption makes it possible to conduct machining when inaccuracies occur (an increase of the process variables monitored), while the required quality is preserved.

While making use of the abovementioned optimization results, a normalization of the values of the process variables was performed on individual fast headstocks. Sample results of the normalization of the acceleration of the vibrations of the turntable on the third fast headstock are presented in Fig. 3.

Table 1. Example results of the optimization of the division of allowance of fast headstocks with rough and form grinding

| fast head | limiting values | | | | | | „safe” values | | | | | |
|-----------|-----------------|----------------------|-----------------------------|-----------------------------|------------|-------------------------------|---------------|----------------------|-----------------------------|-----------------------------|------------|-------------------------------|
| | v_w mm/s | a μm | Δh μm | Δp μm | F_n N | \dot{v}_s m/s^2 | v_w mm/s | a μm | Δh μm | Δp μm | F_n N | \dot{v}_s m/s^2 |
| 1 | 6.2 | 148 | 31 | 4.9 | 38 | 2.7 | 4.4 | 139 | 20 | 3.3 | 30 | 2.2 |
| 2 | 6.2 | 97 | 27 | 4.5 | 36 | 2.6 | 4.4 | 100 | 18 | 3.1 | 29 | 2.1 |
| 3 | 6.2 | 56 | 19 | 3.5 | 30 | 2.3 | 4.4 | 60 | 14 | 2.5 | 24 | 1.9 |

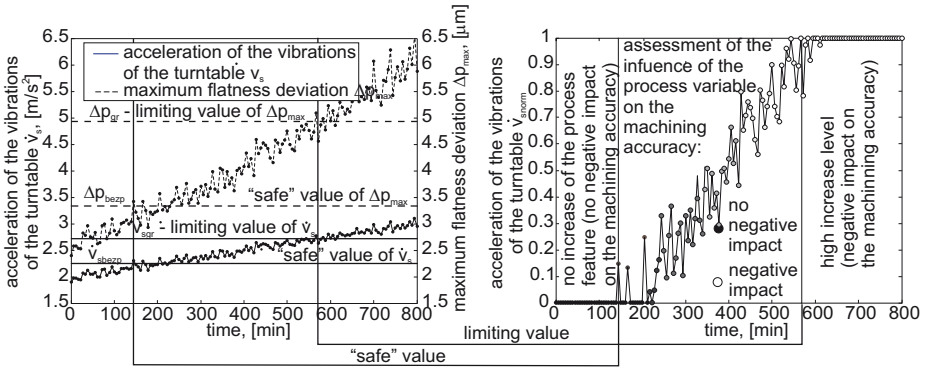


Fig. 3. An example of the normalization of the acceleration of the vibrations of the turntable. The normalized variable value corresponds to the level of its influence of the flatness deviation of the elements machined.

As a result, a diagnostic variable was obtained (a normalized value of the process variable monitored), which constitutes the assessment of the level of its negative impact on the machining accuracy:

$$F_{norm}, \dot{v}_{snorm} = \begin{cases} 0 & \text{no increase of the process feature monitored} \\ (0, 1) & \text{increase level of the process feature monitored} \\ 1 & \text{high increase level (quality risk)} \end{cases} \quad (4)$$

Exceeding of the limiting values of the process variables determined by optimization, results in the generation by fuzzy models of inaccuracies of the diagnostic signal of the value = 1, which indicates a high inaccuracy level in the process, which might result in exceeding of the accepted share of defective products in the overall number of the elements machined.

4 Machining Accuracy Assessment

In the simplest case, an assessment of the correctness of the machining performed involves making a decision as to whether the process monitored leads

Table 2. Fuzzy relationships which describe the state of machining conducted in the case of a divalent assessment

| rule no. | F_{nnorm} | operator | \dot{v}_{snorm} | machining quality assessment |
|----------|-------------|----------|-------------------|---|
| 1 | if small | and | small | objects with required accuracy class |
| 2 | if large | or | large | objects outside required accuracy class |

to obtaining products which meet the quality criteria, and so whether the machining is conducted properly, or whether as a result of machining we obtain products which are outside the required quality class (Table 2). Such a decision concerning the correctness or incorrectness of the machining process was made on the basis of the values of the normalized process variables. These variables accept values 1 — when the variable monitored exceeded the limiting value which causes a risk to the achievement of the required product’s quality, or value < 1 — when the variable monitored is in the permissible area, which is the result of the requirements imposed on the machining conditions.

An analysis of the experimental data and the results of the optimization of the parameters of the grinding process in question [9] allows to make conclusions about a smaller influence of the increase of the component of the normal grinding force F_n in relation to the increase of the vibration accelerations of turntable \dot{v}_s on the quality parameters achieved. An increase of the normal component, while causing deformations, leads to the increase of the height deviation. This parameter, in the case of ceramic elements machined (in this case, used in valve sealing) can be characterised by a significant deviation. The flatness deviation is a more important criterion in the case of elements which mate in a shoe manner. An excess of this type of deviation constitutes the basis of a disqualification of the product.

While making use of the abovementioned analyses and other experiments from this area [9], [10], a system of a multilayer assessment of the machining quality was developed (Fig. 4), which differentiates the influence of diagnostic

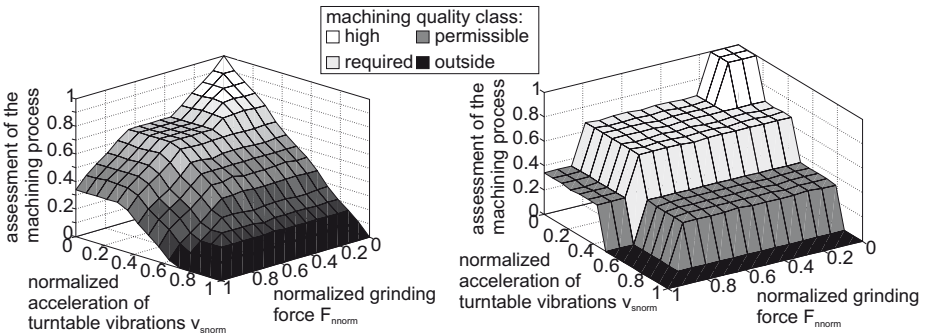


Fig. 4. The reply space of the fuzzy model of the multilayer quality assessment of the machining process: a) continuous assessment, b) jump assessment

Table 3. Fuzzy relationships which describe the state of machining in the case of a multilayer assessment

| rule no. | F_{norm} | operator | \dot{v}_{snorm} | machining quality assessment |
|----------|-------------|----------|-------------------|------------------------------|
| 1 | if small | and | small | high |
| 2 | if small | and | medium | required |
| 3 | if small | and | large | permissible |
| 4 | if medium | and | small | required |
| 5 | if medium | and | medium | required |
| 6 | if medium | and | large | permissible |
| 7 | if large | and | small | required |
| 8 | if large | and | medium | required |
| 9 | if large | and | large | permissible |
| 10 | if limiting | and | small | permissible |
| 11 | if limiting | and | medium | permissible |
| 12 | if limiting | or | limiting | outside quality class |

variables (the normalized grinding force F_{norm} and the normalized acceleration of turntable vibrations \dot{v}_{snorm}). The fuzzy relationships which describe the state of machining performed in the case of a multilayer assessment were presented in Table 3.

The process quality assessment strategy applied, allows one to conduct it on the basis of the normalized values of the process variables. The fuzzy systems performs an assessment of the level of the growth of individual variables and of their influence on the change of the machining accuracy. The classification conducted can have a multivalent form and can correspond with the product’s quality classes.

By a simple modification of linguistic rules, it is possible to differentiate a negative impact on the substantial parameters of the product’s quality. This contributes to a significant universality of the assessment method applied with regard to the machining correctness. It can be used in the assessment of other technological processes.

5 Conclusions

The value of the process variables monitored, which enable an assessment of the machining accuracy, depends on the accepted parameters and conditions of machining. An application of the theory of fuzzy sets enables such a normalization of the monitored values of process variables, which takes into account the level of its impact on the substantial parameters of the product’s quality.

A change of the assessment index of the product’s quality in time, provides information on the level of an unfavourable impact of inaccuracies on the substantial parameters of the product’s quality. A change of its index can be of a temporary nature caused by a change of the allowance of the elements ground, or of a global nature, connected with the wear of the grinding wheel.

This constitutes the basis for the identification of the reasons of the occurrence of inaccuracies in the process of automatic grinding of ceramic elements.

An assessment of the results of a total influence of inaccuracies on the results of the grinding process leads on to apply effective procedures in the processing of incomplete, inexact and uncertain data based on the theory of fuzzy sets. An introduction of the theory of fuzzy sets enables the algorithms applied and the machining quality assessment with the use of linguistic rules, to be made universal. These methods also allow the process operator's expert knowledge to be made formal. Such an operator, while making use of heuristics only, is able to make a correct diagnosis concerning the state of the process monitored.

References

1. Malkin, S., Hwang, T.W.: Grinding Mechanisms for Ceramics, *Annals of the CIRP* Vol. 45, No.1, (1996), pp. 569-580
2. Shaw, M.C.: Principles of Abrasive Processing, Oxford University Press, New York (1996)
3. Kitajima, K., Cai, G.Q., Kumagai, N., Tanaka, Y., Zheng, H.W.: Study on Mechanism of Ceramics Grinding, *Annals of the CIRP* Vol. 41, No. 1, (1992), pp. 367-370
4. Zhang, B., Howes, T.D.: Material - Removal Mechanisms in Grinding Ceramics, *Annals of the CIRP* Vol. 43, No. 1, (1994), pp. 305-308
5. Kacalak, W.: New Methods of Fine Grinding Small Ceramics Elements, VII Scientific School of Abrasive Machining, Gdask, (1984), pp. 53-64
6. Piegat, A.: Fuzzy Modelling and Control, EXIT, Warsaw (1999)
7. Tsoukalas, L.H., Uhrig, R.E.: Fuzzy and Neural Approach in Engineering, John Wiley & Sons, New York (1997)
8. Byrne, G., (ed.): Tool Condition Monitoring (TCM) - The Status of Research and Industrial Application, *Annals of the CIRP* Vol. 44 No. 2, (1995), pp. 541-563.
9. Lipiński, D.: Quality Supervising in Grinding Processes of Small Ceramics Elements using Artificial Intelligent Methods. PhD Thesis, Koszalin University of Technology (2005).
10. Kacalak, W. (ed.): Intelligent System of Inaccuracy Minimization and Disturbances Compensation in Grinding Processes. State Committee for Scientific Research (2001).

A Dynamic Resource Broker and Fuzzy Logic Based Scheduling Algorithm in Grid Environment

Jiayi Zhou¹, Kun-Ming Yu², Chih-Hsun Chou², Li-An Yang², and Zhi-Jie Luo²

¹ Institute of Engineering Science, Chung Hua University

² Department of Computer Sciences and Information Engineering,

Chung Hua University

Hsin-chu, Taiwan 300, ROC

{jyzhou, laving, kevin}@pdlab.csie.chu.edu.tw,

{yu, chc}@chu.edu.tw

Abstract. Grid computing is a loosely couple distributed system, and it can solve complex problem with large-scale computing and storage resources. Middleware plays important role to integrate heterogeneous computing nodes. Globus Toolkit (GT) is a popular open source middleware to build grid environment. However, a job submission has lots of complicate operations in GT especially in a large scale grid. Moreover, the information discovery component of Globus Toolkit can only provide the summarized information from Grid Head instead of each computing node. Furthermore, job scheduling is another important issue in the high performance Grid computing. An appropriate scheduling algorithm can efficiently reduce the response time, turnaround time and increase the throughput. In this paper, we develop a resource broker module for GT infrastructure, which can dynamically describe and discover the resource information of computing nodes. Moreover, we design an adaptive fuzzy logic scheduler, which utilizes the fuzzy logic control technology to select the most suitable computing node in the Grid environment. For verifying the performance of the proposed scheduling algorithm, we also implement a resource broker as well as fuzzy logic scheduler based on Globus Toolkit 4. The experimental results show our algorithm can reduce the turnaround time compared with round-robin and random dispatching methods. The experiments also show that our algorithm has better speed-up ratio than round-robin and random dispatching when number of computing nodes increasing.

1 Introduction

Grid computing and pervasive computing can solve complex problems with large-scale computing and data resources. Grid is a computing architecture based on internet connection [5], and it shares heterogeneous computing and storage resources with internet and has higher scalability. Instead of traditional Cluster system, it can easily add more computing resource with lower cost. Middleware applications play a significance role in grid environment [2, 8, 9], it enabled distributed computers communicate to each other with different type of CPUs, OS, and various hardware

specification. It acts as a broker between user and provided resources. Globus Toolkit (GT) is one of popular middleware for building grid computing. GT lets people share computing power, database, and other tools securely across network connections. GT also includes software for security, information infrastructure, resource management, data management, communication, fault detection, and portability.

Although GT seems very convenient to achieve the pervasive computing [13, 15], however, there are two drawbacks when submits a job. First, to submit a job to a computing grid using Globus Toolkit needed many complicated procedures as well as lots of command-line operations. Second, grid system is a heterogeneous computing environment, how to submit jobs to computing nodes with most available computing resource is an important scheduling problem.

Job scheduling is another important issue in grid computing. An inappropriate scheduling scheme will increase the response time, turnaround time and decrease the throughput. The scheduling scheme can be categorized into two types: static and dynamic. The static scheme dispatches jobs into computing nodes without considering the workload of each node when jobs arrived, for example, random selection and round robin selection. On the other hand, the dynamic scheme can dispatch jobs with currently workload information when jobs to be executing. It will gather the CPU utilization, CPU run queue length, memory usage, and memory swapping as indexes to identify the workload of each computing node.

Conventionally, scheduling algorithm usually uses some fixed threshold value (Crisp Set) to distinguish the loading state of each computing node. However, these values can not correctly respond the actual workload of computing node [11]. To solve this problem, we propose a Fuzzy Logic Scheduler (FLS) algorithm to estimate workload of each node by fuzzy logic control. However, the effects of CPU utilization and CPU run queue length to workload is not linear. Each factor has different growing model, for example, the workload grows rapidly when the CPU run queue length greater than threshold. An unsuitable measuring module can not reflect the correct workload.

A resource broker and scheduler are fundamental for any large-scale grid computing [10]. In this paper, we proposed and implemented a resource broker module and fuzzy logic scheduler module mainly targeted for Globus Toolkit infrastructure. Rather than the grid resource information service (GRIS) in MDS, the resource broker monitor can gather the detail information of each computing node, not the summarized information.

The rest of this paper is organized as follows. In section 2, some preliminaries about grid middleware Globus Toolkit and fuzzy logic control are given. Proposed resource broker module and fuzzy logic scheduler module are described in section 3. Section 4 is experimental results, and section 5 is our conclusions.

2 Background and Motivation

2.1 Globus Toolkit

The Globus Toolkit [6, 7, 12] is a set of components which includes basic service [3, 4] for security, information, resource management, storage and communication. Each service is distinct and has well defined interfaces so they can be incorporated into

applications or tools. Globus Toolkit 4 (GT4) is an open source software toolkit which contains many standard components. In GT4, monitoring and discovery services (MDS) are mainly concerned with the collection, distribution, indexing, archival, and it provide grid information such as the resources that are available and the status of the computing nodes. This information may include number of processors available, CPU utilization, network utilization, storage, memory usage, and swap usage of each node. Reliable file transfer (RFT) provides a web service interface for transfer and deletion of files. RFT receives requests via SOAP message over HTTP and utilizes GridFTP [1]. RFT also uses a database to store the list of file transfers and their states. Moreover, Globus resource allocation manager (GRAM) is an integral component of a computation grid. It is responsible for processing job-requests, enables remote monitoring of jobs.

The related components for job submission in GT4 is shown in Fig. 1. GT4 Job Submission Components

In addition, the problem of selecting appropriate computing nodes is not trivial. User needs query information from MDS of Grid Head A to Grid Head N. Afterward, user collects the results and selects the jobs manually, then send the jobs and write a RSL to execute jobs. Moreover, MDS is unable to obtain the information of sole computing node. It can only get the summarized information, for example, Grid Head A contains 4 computing node each has 1 GHz CPU power and 256 MB memory (Level 3 Information). However, MDS only can report Grid A has 4 GHz CPU power and 1 GB memory (Level 2 Information). There are too many complicated operations for job submission when scaling of grid is large [14].

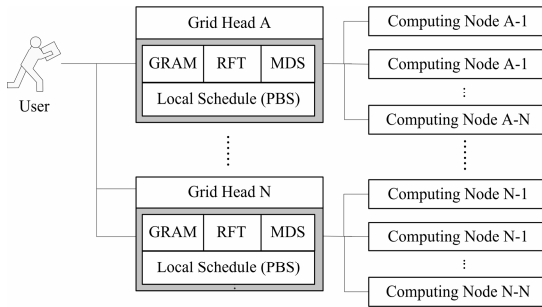


Fig. 1. GT4 Job Submission Components

In order to solve this problem, we design and implement a Resource Broker (RB) module in GT4. Main goal of the resource broker module is to eliminate the overelaborate job submission procedure and obtain more detail of resource information of each computing nodes.

2.2 Job Scheduling and Workload Measurement

Job scheduling can be classified into two categories: dynamic and static. A static job scheduling scheme dispatches jobs to the computing node without considering the current workload of each node, for example, round-robin and random dispatching

method. It has least overhead and it is proper for tightly coupled distributed system with the identical jobs. However, grid is a loosely coupled distributed system and it is also a heterogeneous computing environment. A dynamic job scheduling scheme is more appropriate for grid environment because it dispatches jobs according to the current workload status of each node. However, there are many different kinds of operating system, CPU, memory size, network bandwidth in a grid. How to measure the workload of each computing nodes by using these resource information is a difficult problem. There are many conventional jobs scheduling algorithms adopt fixed threshold value (Crisp Set) to determine the workload of computing nodes. However, there are many resource information (variables) corresponding to the workload and sole threshold value with IF-THEN method typically can not represent the real workload status of each computing nodes. The fuzzy logic control has a better adaptability, robustness, and fault tolerance for these conditions. Therefore, we propose a Fuzzy Logic Scheduler (FLS) algorithm to evaluate the workload of each computing nodes in a grid.

3 Proposed Resource Broker Module and Fuzzy Logic Scheduler Module

In this section, we will describe the Resource Broker Module and Fuzzy Logic Scheduler Module in detail. By applying the proposed modules, we can correctly evaluate the workload status of every computing nodes(level 3) and decrease the tedious operations for job submission in GT4.

3.1 Resource Broker Module

The overall structure of the proposed resource broker module is shown in Fig. 2. The resource broker can reduce the complexity of user operations especially in large-scale grid system. The resource broker will discovery the information of each level 3 computing node. The workload of each nodes can be calculated more precise with the CPU utilization, CPU run queue length, etc..., which can not obtained via Globus Toolkit. The procedure of job submission with resource broker module is as follows:

- Step 1:** User prepares their executable jobs and input data and then describes the required resource specification of computing node. Submitting jobs to Resource Broker (RB) via Command Line User Interface (CLUI).
- Step 2:** RB uses Information Manager to connect to MDS then gets the available computing nodes.
- Step 3:** RB invokes Resource Broker Monitor (RBM) to retrieve the resource information of computing nodes, which including CPU utilization, CPU run queue length, total and used memory, total and used swap, etc.
- Step 4:** Control Center (CC) receives the loading information of each node.
- Step 5:** CC sends loading information to Workload Measurement Module (WMM), which utilizes Fuzzy Logic Scheduler (described in 3.2), then get the Node Utilization Score.

- Step 6:** RM selects the appropriate computing nodes according to the Node Utilization Score followed by invoking Transfer Manager (TM) to send jobs to RFT of each selected computing node.
- Step 7:** RB uses Execution Manager to send job execution information to GRAM.
- Step 8:** Finally, RB collects the execution results of jobs then send back to user via CLUI.

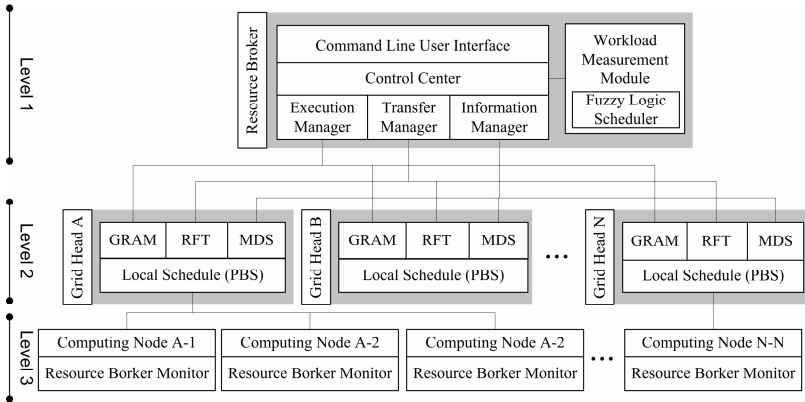


Fig. 2. Globus Toolkit with Resource Broker Module

3.2 Fuzzy Logic Scheduler Module

An appropriate scheduling algorithm can efficiently reduce the response time and turnaround time, and increase the throughput. The workload of each computing node influences upon many factors such as CPU utilization, CPU run queue length, memory utilization, swap utilization, etc. All of them can be categorized into the utilizations of CPU and memory. In this study, a fuzzy logic based scheduler algorithm by referring both utilizations was developed and is described in the following.

3.2.1 Define the Input and Output Fuzzy Variables

As stated in the above, two types of information include the CPU and memory utilizations were used as the workload indices. The CPU utilization contains two factors, the degree of CPU utilization and the degree of run queue length. The memory utilization also contains two factors, the degree of memory utilization and the degree of swap utilization. Since it is difficult to construct fuzzy control rules using four input variables, both pairs of factors were combined to form two input fuzzy variables, the CPU utilization and the memory utilization. It is straightforward to compute the CPU utilization by averaging the degrees of CPU utilization and run queue length, however, through our observation, when the run queue length is distinctness the CPU run queue length affects the workload more significant than the CPU utilization. It means that a nonlinear combination of these two factors required exhibiting a better efficiency. According to our experience, the CPU run queue length affects the workload more significant than CPU utilization when the run queue length grater then five. So we construct the nonlinear mapping in the following.

$$q(ql) = \begin{cases} \frac{1}{1 + \exp[w \cdot (a - ql)]^{\text{deg}}}, & \text{if } ql < a \\ 1 - \frac{1}{1 + \exp[w \cdot (ql - a)]^{\text{deg}}}, & \text{if } ql \geq a \end{cases} \tag{1}$$

$$\text{cpu}(uti, ql) = \sqrt{w_1 \cdot q(ql) + w_2 \cdot (uti / \text{inf})^3} \tag{2}$$

in which *uti* and *ql* represent the degrees of CPU and membership utilizations and *deg*, *a*, *w*₁ and *w*₂ are parameters for tuning the nonlinearity. In this study, the five parameters used in (1) and (2) were set as *deg* = 1.5, *a* = 0.6, *w* = 0.6, *w*₁ = 0.7, and *w*₂ = 0.3. Fig. 3. Function used to form the input variable shows the hyperplane formed by eqs. (1) and (2). It can be found that the curve in the axis of degree of CPU utilization is nearly linear, and is highly nonlinear in the other axis. The definition of the second input variable is analogue to the first one, in which the degree of swap utilization plays a similar role as the degree of run queue length. There are a wide selection to choose your favorite membership function. For example, the triangle and trapezoid function. In this paper, we choose triangle function. The membership functions for both input fuzzy variables are defined in the following

$$\text{Light}(x) = \begin{cases} 1, & \text{if } x < 0 \\ 1 - 2x, & \text{if } 0 \leq x \leq 0.5 \end{cases} \tag{3}$$

$$\text{Medium}(x) = \begin{cases} 2x, & \text{if } 0 \leq x < 0.5 \\ 2 - 2x, & \text{if } 0.5 \leq x \leq 1 \end{cases} \tag{4}$$

$$\text{Heavy}(x) = \begin{cases} 2x - 1, & \text{if } 0.5 \leq x \leq 1 \\ 1, & \text{if } x > 1 \end{cases} \tag{5}$$

in which *x* denotes the value of the input fuzzy variable. Since the aim of the proposed fuzzy system is to evaluate the workload status, so the output fuzzy variable of the

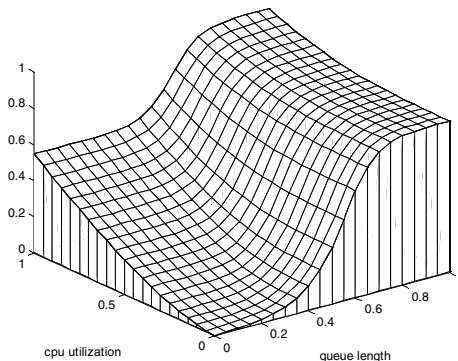


Fig. 3. Function used to form the input variable

system was defined as the workload degree of the computing node. For simplicity, the membership function of the output fuzzy variable was defined the same as the inputs.

3.2.2 Construct the Fuzzy Inference Rules

On the purpose of inferring the loading degree, the fuzzy rules constructed by using the input and output fuzzy variables defined in the previously paragraph are shown in Table 1. In accordance with our experience on loading balance research, the CPU utilization affects the system workload notably then the memory utilization. Therefore, we fill rules by this conception. For example, if the CPU utilization is medium and the memory utilization is light then the loading degree is medium. Opposite to CPU utilization is light and memory utilization is medium then this loading degree is light. By using the Max-Min inference method and the center-of-gravity defuzzification process, the load degree of each computing node can be obtained, with which the scheduling job can be achieved.

Table 1. Fuzzy inference rules

| | | | |
|-----------------|-------------------|---------------|-------------------|
| <i>Memory \</i> | <i>Light</i> | <i>Medium</i> | <i>Heavy</i> |
| <i>Light</i> | <i>Very Light</i> | <i>Medium</i> | <i>Heavy</i> |
| <i>Medium</i> | <i>Light</i> | <i>Medium</i> | <i>Heavy</i> |
| <i>Heavy</i> | <i>Medium</i> | <i>Heavy</i> | <i>Very Heavy</i> |

4 Experimental Results

To evaluate the performance of the proposed fuzzy logic scheduler module, we have designed a grid computing environment which consist four grid systems by using Globus Toolkit 4, the hardware and software configuration are described in Table 2. Each grid system has one head node and several computing nodes. Three schemes, the proposed fuzzy logic scheduler (FLS), round-robin (RR), and random were executed for performance evaluation. In the experiment, we use Fibonacci sequence benchmarks for comparison. The benchmark of Fibonacci sequence is to compute the given length of it.

Table 2. Experimental Environment

| | Grid A | Grid B | Grid C | Grid D |
|-----------------|---|--------------------------------------|---|--------------------------------------|
| Grid Head | Pentium 3 800MHz 256 MB memory | AMD 1.2GHz SMP 1 GB memory | Pentium 4 2.8GHz 756 MB memory | Xeon 3.2GHz HT * 2 1 GB memory |
| Computing Nodes | Pentium 3 800MHz 256 MB memory | AMD XP 1.4GHz 768 MB memory | AMD XP 1.4GHz 768 MB memory | Pentium 4 3GHz HT 1 GB memory |
| Number of Nodes | 9 | 3 | 4 | 4 |
| Software | Debian 3.1r2; Globus Toolkit 4.0.2; DRBL 1.7.1; GCC 3.3.5 | | | |

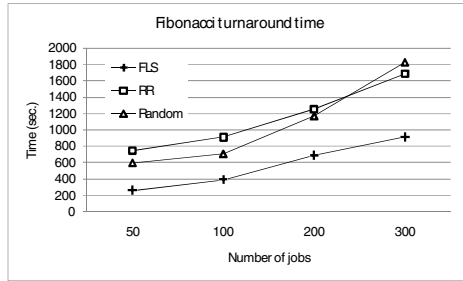


Fig. 4. Performance comparison of turnaround time with difference number of jobs

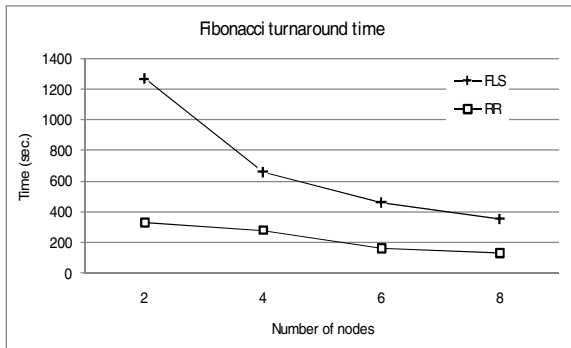


Fig. 5. Performance comparison of turnaround time with difference number of nodes

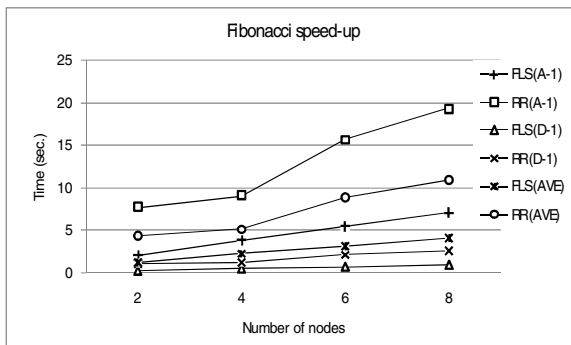


Fig. 6. Performance comparison of Speed-up ratio with difference number of nodes

Figure 4 to Figure 6 are the simulation results of Fibonacci sequence. The turnaround time with different number of jobs is shown in Figure 4. Here we can observe the turnaround time is decreased for FLS compared to RR or random dispatching methods. The turnaround time with different number of nodes is shown in Figure 5, it illustrates FLS can reduce turnaround time when number of nodes grown. To compare the speed-up ratio, we choose one computing nodes from Grid A and one

computing nodes Grid D for comparison. For example, four computing nodes means select two nodes from Grid A and two nodes from Grid D. Afterward, we can get the speed-up ratio for FLS and RR relative to single node of Grid A or Grid D or average of them. The results are depicted in Figure 6. In Figure 6, FLS (A-1) denotes the speed-up ratio of FLS compared with single node in Grid A. Similarly, FLS (ave) denotes the speed-up ratio of FLS compared with average turnaround time which choosing one node from Grid A and one node from Grid D. The results show that FLS always has better performance than RR compared to single node of Grid A, Grid D, and average of them.

5 Conclusions

In this paper, we design and implement a resource broker module for Globus Toolkit. The resource broker can collect the precise resource information of each computing nodes from resource broker monitor and can execute the job submission procedures automatically. Moreover, within the proposed resource broker module, we utilize the concept of fuzzy logic to correctly evaluate the workload of each computing nodes in a grid. Since there are too many variables to determine the real workload of computing nodes, our proposed fuzzy logic scheduler can effectively calculate the current workload of each computing nodes in a grid. In addition, it can help the job scheduler making an appropriate decision. For verifying the performance of our proposed resource broker module, we also implement a grid computing by using Globus Toolkit 4. Three schemes, the proposed fuzzy logic scheduler (FLS), round-robin (RR), and random were implemented for comparison. The experimental results show that our proposed scheduling algorithm can successfully reduce the turnaround time. Furthermore, the speed-up ratio of our algorithm is better than round-robin in every simulation.

References

1. Allcock, W., "GridFTP: Protocol Extensions to FTP for the Grid," Global Grid Forum GFD-RP.020, (2003).
2. Berman, F., Fox, G. and Hey, T., "Grid Computing: Making the Global Infrastructure a Reality," John Wiley & Sons, (2003).
3. Foster, I., Berry, D., Djaoui, A., Grimshaw, A., Horn, B., Kishimoto, H., Maciel, F., Savva, A., Siebenlist, F., Subramaniam, R., Treadwell, J. and Reich, J.V., "Open Grid Services Architecture V1," (2004).
4. Foster, I. and Kesselman, C. Globus, "A Metacomputing Infrastructure Toolkit," International Journal of Supercomputer Applications, vol. 11, no. 2, pp.115-129, 1998.
5. Foster, I., Kesselman, C., "The Grid: Blueprint for a New Computing Infrastructure," Morgan Kaufmann, (1998).
6. Foster, I., Kesselman, C., Nick, J. and Tuecke, S., "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration," Open Grid Service Infrastructure WG, Global Grid Forum, (2002).
7. Foster, I., Kesselman, C., Nick, J.M. and Tuecke, S., "Grid Services for Distributed Systems Integration," IEEE Computer, vol. 35, no. 6, (2002) pp. 37-46

8. Foster, I., Kesselman, C. and Tuecke, S., "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *International Journal of Supercomputer Applications*, vol. 15, no. 3, (2001) pp. 200-222
9. Foster, C. Kesselman, J. Nick, and S. Tuecke, "Grid services for distributed system integration," *IEEE Computer*, vol. 35, no. 6, (2002) pp. 37-46
10. Fujimoto, N., Hagihara, K., "A Comparison among Grid Scheduling Algorithms for Independent Coarse-Grained Tasks," *Symposium on Applications and the Internet-Workshops*, (2004) pp. 674-680
11. Huang, J., Jin, H., Xie, X., Zhang, Q., "An approach to grid scheduling optimization based on fuzzy association rule mining," *First International Conference on e-Science and Grid Computing*, (2005) pp. 189-195
12. Lin, A., Maas, P., Peltier, S. and Ellisman, M., "Harnessing the Power of the Globus Toolkit," *ClusterWorld*, vol. 2, no. 1, (2004).
13. Open Grid Services Architecture Data Access and Integration (OGSA-DAI) Project. Available on <http://www.ogsa-dai.org.uk>.
14. Rong, H., Zhigang, H., "A Scheduling Algorithm Aimed at Time and Cost for Meta-tasks in Grid Computing Using Fuzzy Applicability," *Eighth International Conference on High-Performance Computing in Asia-Pacific Region*, (2005), pp. 564-569.
15. Sample, N., Keyani, P., Wiederhold, G., "Scheduling Under Uncertainty: Planning for the Ubiquitous Grid," *Int. Conf. on Coordination Models and Languages*, (2002) pp. 300

Improving Business Failure Predication Using Rough Sets with Non-financial Variables

Jao-Hong Cheng¹, Chung-Hsing Yeh^{2,3}, and Yuh-Wen Chiu^{1,4}

¹ Department of Information Management, National Yunlin University of Science and Technology, Douliou, Yunlin, 640, Taiwan
jhcheng@yuntech.edu.tw

² Clayton School of Information Technology, Faculty of Information Technology, Monash University, Clayton, Victoria 3800, Australia
ChungHsing.Yeh@infotech.monash.edu.au

³ Department of Transportation and Communications Management, College of Management, National Cheng Kung University, Tainan, Taiwan
monash@mail.ncku.edu.tw

⁴ Department of Information Management, Far East University, Tainan, Taiwan
g9020823@yuntech.edu.tw

Abstract. Rough set models with financial variables have proven to be effective in predicting business failure. To enhance the predictive performance of rough set models, this paper includes a non-financial variable, auditor switching, into the modeling process, in addition to 14 financial ratios commonly used in business failure research. An empirical study on 62 failed firms and 62 one-to-one matching non-failed firms in Taiwan between 1998 and 2005 is conducted, using available data for the three years before failure. Six rough set models are constructed individually with and without the auditor switching variable, using the three-year data respectively. The empirical study shows that the non-financial variable is the most significant attribute and plays an essential role in enhancing the performance of rough set models. These findings highlight the effectiveness of rough set models for business failure prediction and particularly the importance of incorporating non-financial variables in business failure research.

1 Introduction

Business failure has been a worldwide economic and social problem, and its prediction has become a major issue in the corporate finance research. Business failure research starts from Altman's pioneering work using multivariate discriminant analysis with a set of five financial ratios for distinguishing failed firms from non-failed firms [1]. To improve the prediction accuracy by overcoming shortcomings of the discriminant analysis model, other cross-sectional statistical models have been developed, such as logit analysis [2] and probit regression models [3]. The advances in computational intelligence techniques have led to the development of intelligent models for analyzing and predicting business failure, including decision trees [4], survival

analysis [5], expert systems [6], neural networks [7], [8], rough sets [9]-[11] and hybrid models [12], [13]. With the aim of providing a higher level of prediction accuracy, these non-parametric models need not to pre-specify a functional form, nor the distribution assumptions of model variables and errors [11]. In particular, rough set models have demonstrated their advantages over other models for business failure prediction [10].

Based on rough set theory developed by Pawlak [14], rough set models have proved to be an effective tool for analyzing financial information systems involving a set of firms (objects) characterized by a set of multi-valued financial ratios and qualitative variables [12]. As an approach to handling uncertainty and vagueness of information in classification analysis, rough set models perform complete classification of objects, by generating a set of decision rules from a set of examples. The generation of rules often involves the generation of reducts or a core of attributes, which are particular subsets of attributes that provide the same classification quality as the full set of attributes. As such, a rough set model has an advantage over a functional or relational model, as it explains the preferential attitude through easily understandable rules with significant attributes (those in a reduct) only.

To reflect the fact that the patterns of object classes often overlap in real decision making settings, variable precision rough set models which incorporate probabilistic decision rules are commonly used [15]. Existing models for predicting business failure are based on a set of financial ratios, which are regarded as objective (quantitative) indicators of business failure. To further examine if non-financial factors may play a significant role in business failure prediction, we construct rough set models using a non-financial (qualitative) variable together with 14 financial ratios, as the rough set approach can handle both quantitative and qualitative attributes. As such, one departure point of this paper is to examine if the development of rough set models can help improve our understanding of business failure research by incorporating non-financial variables. The effectiveness of these models, particularly the significance of the non-financial variable will be examined by an empirical study of 124 Taiwanese firms.

2 Basic Concepts of Rough Sets

In rough set models, knowledge about objects is represented in the form of an information table [16]. The rows and columns of the information table represent objects and attributes respectively, and entries of the table are attribute values. The central concept of rough sets is a collection of rows that have the same values for one or more attributes, which form the indiscernibility relation about the finite set of objects (called the universe). Any set of all indiscernible objects is called an elementary set, which forms a basic granule of knowledge about the universe. Any subset of the universe can be expressed either precisely or roughly. A set of objects is said to be crisp (precise) if it is a union of some elementary sets; otherwise, the set is rough (imprecise). A rough set can be represented by a pair of crisp sets, called the lower and upper approximations. For a subset of the universe, the lower approximation consists of all objects that certainly belong to the set and the upper approximation includes objects that possibly belong to the set. The difference between the upper and lower

approximations constitutes the boundary region of the vague concept (a subset of the universe). Approximations are two basic operations in rough set theory.

Decision rules of rough set models constitute a formal language to describe approximations in logical expressions (implications) [17]. Decision rules are expressed in the form of “If <conditions> Then <decisions>”. Certain rules correspond to the lower approximation, whereas the uncertain rules correspond to the boundary region. The certainty and the coverage factors of decision rules are conditional probabilities which describe how exact our knowledge is about the universe. Each decision rule is characterized by the strength of its conclusion, which is indicated by the number of objects satisfying the condition part of the rule and belonging to the decision class [10]. In generating decision rules based on inductive learning principles, the objects are regarded as examples of decisions. To induce decision rules for describing a set of objects, the examples belonging to it are called positive and all the others negative. A decision rule is discriminant if it distinguishes positive examples from negative ones and it is minimal. With the prescription ability of how to make a decision under specific conditions, decision rules derived give pertinent information useful for decision support.

To have the best quality of approximation of classification with a minimal set of decision rules, not all the condition attributes in the information table are to be used. An important step in the rough set approach is to identify the minimal subset of condition attributes (called a *reduct*), which provide the same quality of classification as the whole set of attributes. Attributes that do not belong to a reduct are superfluous in terms of classification of elements of the universe [16]. If an information table has more than one reduct, the intersection of all reducts is called the *core* of attributes. The core is a collection of the most significant attributes in the table, without any of which the quality of classification will reduce.

3 Financial Ratios and Non-financial Variables for Business Failure Analysis

Financial ratios have been commonly perceived as having the ability to predict the failure of business. A large number of ratios have thus been proposed in the literature. For example, Curtis [18] identifies 79 financial ratios for analyzing corporate performance and structure, including 28 ratios useful for predicting various forms of corporate difficult, 34 additional ratios from contemporary textbook literature, and 17 ratios used in specific studies or organizations. To represent corporate financial phenomena which express the relationships between the ratios and the characteristics that they purport to reveal, these financial ratios are grouped into three categories: (a) profitability, (b) managerial performance, and (c) solvency. The most important financial ratios are in the solvency category (e.g. working capital/total assets, total debt/assets), and the next is in the profitability category, suggesting that the viability of a business depends on profit making to a large extent [19]. In addition, the performance and survival of a business are influenced by several factors, including environment, national and international economic situations. As such, additional financial ratios have been proposed for inclusion in the business failure analysis, including economic variables [20].

In business failure prediction studies using rough sets, workable models normally use 10 to 86 if-then decision rules with 2 to 5 variables (financial ratios) [13]. A total of 13 different financial variables are used in these models and some of these variables are quite similar as they are surrogates for the same underlying economic factors. Although rough set models with financial variables may produce good predictive ability of business failure, business's performance and future may be influenced by qualitative, non-financial characteristics, such as strategic criteria [21], social importance of the firm, and the strength of its bank relationship [22]. Based on this notion and data availability, the rough set models to be developed in this study consider a set of 15 variables as condition attributes, including 14 financial ratios and one non-financial variable.

The 14 financial ratios cover all the categories suggested by previous studies, including (a) solvency (current ratio, A_1 ; quick ratio, A_2 ; liabilities/assets ratio, A_3 ; times interest earned ratio, A_4), (b) managerial performance (average collection turnover, A_5), (c) profitability (return on total assets, A_6 ; return on shareholders' equity, A_7 ; operating income to paid-in capital, A_8 ; profit before tax to paid-in capital, A_9 ; earnings per share, A_{10}), (d) financial structure (shareholder's equity/total assets ratio, A_{11}), and (e) cash flow (cash flow ratio, A_{12} ; cash flow adequacy ratio, A_{13} ; cash flow reinvestment ratio, A_{14}). The non-financial variable considered is the status of auditor switching (A_{15}), which is used to indicate whether or not a firm had changed its auditor in the past one, two or three years before failure. Previous studies have suggested that failing firms are more likely to switch their auditor up to three years before failure, mainly resulting from disputes between auditors and managers over accounting methods as well as disagreements on audit opinions and qualifications [23].

4 Rough Set Models for Business Failure Prediction

4.1 The Data

All the data required were collected from the Taiwan Economic Journal, a local database including annual reports of more than 4,500 firms in Taiwan and in China. 62 firms which failed between 1998 and 2005 were identified. In the context of this empirical study and according to the Operational Rules of The Taiwan Stock Exchange Corporation, a firm is said to be failed if it (a) was filed for bankruptcy, (b) was placed into suspension of trading, (c) was altered trading method, or (d) was involved in unusual actions such as a cease trading order or delisting. To match one by one with the 62 failed firms, 62 non-failed or healthy firms were selected. The 62 non-failed firms were selected randomly among those in the same industry with similar total assets to the corresponding failed firms. The non-failed firms must have financial data available for the same fiscal years that were used for the corresponding failed firms. As such, a total sample size of 124 firms was used, which is comparable to existing studies, e.g. 80 [10], 90 [11], and 60 [12]. The use of one-to-one match of failed and non-failed firms is consistent with business failure prediction studies [10]-[12], [19].

Based on the 15 variables identified in the previous section, the corresponding financial ratios and auditor switching data of these 124 firms were collected for a period of three years, starting from year -3 (three years before failure) and ending with year -1 (one year prior to the year of failure, the last year that the firm was in business). Clearly the actual year of failure (year 0) is not the same for those 62 failed firms, as they did not all fail in the same year. The condition attributes of the rough set models are to be selected from the 15 variables, while the only decision attribute of the models is for indicating whether a firm has failed. The value of the decision attribute is 0 if the firm has failed and 1 if not failed.

The rough set analysis of the data collected for the 62 failed and 62 non-failed firms was performed using the ROSE (Rough Sets Data Explorer) system. With the three-year data collected, three rough set models were constructed respectively, each with its own set of decision rules. These three rough set models were constructed and evaluated in four phases: (a) preprocessing – preliminary data analysis and modifications, including discretization; (b) reducts – methods dealing with the reduction of attributes; (c) rules – methods used to generate decision rules; and (d) classification – validation of decision rules. In order to facilitate model construction and testing, 33 firms failed in the years from 1998 to 2001 and 33 matching non-failed firms were selected as a training (learning) sample for generating decision rules of rough set models. The remaining 29 failed and 29 non-failed firms were used as a testing (holdout) sample.

4.2 Data Processing

The data collected were first pre-processed to form the information table, which was the knowledge representation in rough set models. Then the continuous condition attributes were discretized by dividing the original domains of the condition attributes into sub-intervals. Instead of asking human experts, which might be impractical, relatively costly and subjectively [11], we used the ROSE system for data discretization. The discretization process produced 2 or 3 intervals for each of the 14 condition attributes (A_1, A_2, \dots, A_{14}). For example, in the data of year -1 (one year before failure), the first attribute (the current ratio, A_1) had three intervals: interval '0' from -inf to 64.9; interval '1' between 64.9 and 149.8, and interval '2' between 149.8 and +inf.

4.3 Reduct: Core of Attributes

The core of attributes was generated to find out the most meaningful attributes. The attributes in a core are indispensable for discrimination and a non-empty core helps determinate the most important attributes as far as the approximation of classes is concerned [10]. Table 1 shows the attributes in the core for the data sets in year -1, year -2, and year -3 (one, two and three years before failure) respectively. As shown in Table 1, three non-empty cores are generated for all the three-year data respectively, each with a different set of attributes. The non-financial variable (auditor switching, A_{15}) and cash flow ratio (A_{12}) are included in the core of all three data sets, thus being the most significant attributes of business failure.

Table 1. Core of attributes for three data sets

| Data set | Core |
|----------|---|
| Year -1 | A ₁₀ , A ₁₂ , A ₁₅ |
| Year -2 | A ₂ , A ₅ , A ₁₂ , A ₁₃ , A ₁₄ , A ₁₅ |
| Year -3 | A ₁ , A ₂ , A ₄ , A ₈ , A ₁₀ , A ₁₂ , A ₁₃ , A ₁₅ |

4.4 Decision Rules

The minimal covering rule method was used to generate decision rules for each of the three-year data sets using the reduced set of core attributes. A minimal set of 12 rules was derived when the data of year -1 was applied. For the year -2 and year -3 data, the minimal set of rules generated consisted of 19 rules and 24 rules respectively.

4.5 Classification: Validation of Decision Rules

To validate the three rough set models for the three-year data sets, the corresponding sets of decision rules were evaluated first on the training sample, and then on the testing sample. Table 2 summarizes the result of the classification accuracy of three models (for the data sets of year -1, year -2 and year -3 respectively) for both the training and testing samples.

Table 2. Classification accuracy of three models

| | Year -1 | Year -2 | Year -3 |
|------------------------|---------------|---------------|---------------|
| <i>Training sample</i> | | | |
| Failed firms | 33/33 (100%) | 32/33 (97.0%) | 32/33 (97.0%) |
| Non-failed firms | 33/33 (100%) | 33/33 (100%) | 33/33 (100%) |
| Total | 66/66 (100%) | 65/66 (98.5%) | 65/66 (98.5%) |
| <i>Testing sample</i> | | | |
| Failed firms | 18/29 (62.1%) | 21/29 (72.4%) | 20/29 (69.0%) |
| Non-failed firms | 26/29 (90.0%) | 19/29 (65.5%) | 19/29 (65.5%) |
| Total | 42/58 (72.4%) | 40/58 (69.0%) | 39/58 (67.2%) |

As shown in Table 2, the three rough set models produce a high classification accuracy rate for the training sample, although the result for the testing sample is not as good as for the training sample. The overall classification accuracy of the three models constructed is quite satisfactory, which compares favorably with existing rough set models and other business failure prediction models used in previous studies [10], [11]. This result provides clear evidence for the applicability of the rough set approach to the business failure prediction problem.

As indicated in Table 1, auditor switching (A₁₅), along with cash flow ratio (A₁₂), is the most significant attribute in all three models. To further examine the significance of this non-financial attribute, we have applied the rough set approach to the same data sets by considering only the 14 financial ratios as condition attributes. The resulting minimal set of decision rules for the data of year -1, year -2 and year -3 is 13, 20 and 24 respectively. Table 3 summarizes the result of the classification

accuracy of these three sets of decision rules (year -1, year -2 and year -3) for both the same training and testing samples. Comparing the results between Tables 2 and 3 clearly shows that the models with the non-financial variable outperform the models without the non-financial variable. This suggests that a business failure prediction model using financial variables only may not necessarily produce the best result. The inclusion of proper non-financial variables may enhance the performance of the model.

Table 3. Classification accuracy of three models using financial ratios only

| | Year -1 | Year -2 | Year -3 |
|------------------------|---------------|---------------|---------------|
| <i>Training sample</i> | | | |
| Failed firms | 32/33 (97.0%) | 31/33 (94.0%) | 31/33 (94.0%) |
| Non-failed firms | 33/33 (100%) | 33/33 (100%) | 33/33 (100%) |
| Total | 65/66 (98.5%) | 64/66 (97.0%) | 64/66 (97.0%) |
| <i>Testing sample</i> | | | |
| Failed firms | 17/29 (58.6%) | 18/29 (62.1%) | 20/29 (69.0%) |
| Non-failed firms | 24/29 (82.8%) | 18/29 (62.1%) | 17/29 (58.6%) |
| Total | 43/58 (74.1%) | 36/58 (62.1%) | 37/58 (64.0%) |

5 Conclusion

The importance of business failure prediction has been highlighted by a large body of research work. Among numerous methods and models developed to address this issue, the rough set approach has demonstrated its advantages over other approaches in terms of variable assumptions and predictive performances. To provide new insights into the business failure research, we have developed six rough set models by considering a non-financial variable (auditor switching) together with 14 commonly used financial ratios. As exemplified in the empirical study, the non-financial variable is the most significant attribute of business failure and plays an essential role in enhancing the performance of the rough set models. The performance of the models with the non-financial variable is better than the ones using financial variables only. These findings strongly suggest that financial ratios alone may not form a complete set of significant variables for business failure analysis, as conventionally used in existing models. To address the business failure prediction problem effectively, both financial and non-financial factors should be considered.

References

1. Altman, E.: Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy. *Journal of Finance* **4** (1968) 589-609
2. Ohlson, J.A.: Financial Ratios and the Probabilistic Prediction of Bankruptcy. *Journal of Accounting Research* **18** (1980) 109-131
3. Zmijewski, M.E.: Methodological Issues Related to the Estimation of Financial Distress Prediction Models. *Journal of Accounting Research* **24** (1984) 59-82

4. Joos, P., Vanhoof, K., Ooghe, H., Sierens N.: Credit Classification: A Comparison of Logit Models and Decision Trees. Proceedings of the Workshop on Application of Machine Learning and Data Mining in Finance, 10th European Conference on Machine Learning, Chemnitz, Germany (1998) 59-72
5. Luoma, M., Laitinen, E.K.: Survival Analysis as a Tool for Company Failure Prediction. *Omega* **19** (1991) 673-678
6. Messier, W.F., Hansen, J.V.: Inducing Rules for Expert System Development: An Example Using Default and Bankruptcy Data. *Management Science* **34** (1988) 1403-1415
7. Yang, Z.R., Platt, M.B., Platt, H.D.: Probabilistic Neural Networks in Bankruptcy Prediction. *Journal of Business Research* **44** (1999) 67-74
8. Atiya, A.F.: Bankruptcy Prediction for Credit Risk Using Neural Networks: A Survey and New Results. *IEEE Transactions on Neural Networks* **12** (2001) 929-935
9. Slowinski, R., Zopounidis, C.: Application of the Rough Set Approach to Evaluation of Bankruptcy Risk. *International Journal of Intelligent Systems in Accounting Finance and Management* **4** (1995) 27-41
10. Dimitras, A.I., Slowinski, R., Susmaga, R., Zopounidis, C.: Business Failure Prediction Using Rough Sets. *European Journal of Operational Research* **114** (1999) 263-280
11. Beynon, M.J., Peel, M.J.: Variable Precision Rough Set Theory and Data Discretisation: An Application to Corporate Failure Prediction. *Omega* **29** (2001) 561-576
12. Ahn, B.S., Cho, S., Kim, C.Y.: The Integrated Methodology of Rough Set Theory and Artificial Neural Network for Business Failure Prediction. *Expert Systems with Applications* **18** (2000) 65-74
13. McKee, T.E., Lensberg, T.: Genetic Programming and Rough Sets: A Hybrid Approach to Bankruptcy Classification. *European Journal of Operational Research* **138** (2002) 436-451
14. Pawlak, Z.: Rough Sets. *International Journal of Computer and Information Sciences* **11** (1982) 341-356
15. Ziarko, W.: Variable Precision Rough Set Model. *Journal of Computer and System Sciences* **46** (1993) 39-59
16. Pawlak, Z.: Rough Set Approach to Knowledge-Based Decision Support. *European Journal of Operational Research* **99** (1997) 48-57
17. Pawlak, Z.: Rough Sets and Intelligent Data Analysis. *Information Sciences* **147** (2002) 1-12
18. Courtis, J.K.: Modelling a Financial Ratios Categorical Framework. *Journal of Business Finance and Accounting* **5** (1978) 371-386
19. Dimitras, A.I., Zanakis, S.H., Zopounidis, C.: A Survey of Business Failure with an Emphasis on Prediction Methods and Industrial Applications. *European Journal of Operational Research* **90** (1996) 487-513
20. Rose, P.S., Andrews, W.T., Giroux, G.A.: Predicting Business Failure: A Macroeconomic Perspective. *Journal of Accounting, Auditing and Finance* **6** (1982) 20-31
21. Zopounidis, C.: A Multicriteria Decision-Making Methodology for the Evaluation of the Risk of Failure and an Application. *Foundations of Control Engineering* **12** (1987) 45-67
22. Suzuki, S., Wright, R.W.: Financial Structure and Bankruptcy Risk in Japanese Companies. *Journal of International Business Studies* **16** (1985) 97-110
23. Schwartz, K., Menon, K.: Auditor Switches by Failing Firms. *The Accounting Review* **14** (1985) 248-260

Optimization of Fuzzy Model Driven to IG and HFC-Based GAs

Jeoung-Nae Choi¹, Sung-Kwun Oh¹, and Hyung-Soo Hwang²

¹ Department of Electrical Engineering, The University of Suwon, San 2-2 Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea
ohsk@suwon.ac.kr

² School of Electrical Electronic and Information Engineering, Wonkwang University, 344-2, Shinyong-Dong, Iksan, Chon-Buk, 570-749, South Korea
hshwang@wonkwang.ac.kr

Abstract. The paper concerns the hybrid optimization of fuzzy inference systems that is based on Hierarchical Fair Competition-based Genetic Algorithms (HFCGA) and information data granulation. HFCGA is a kind of multi-populations of Parallel Genetic Algorithms (PGA), and it is used for structure optimization and parameter identification of fuzzy model. The granulation is realized with the aid of the Hard C-means clustering (HCM). The concept of information granulation was applied to the fuzzy model in order to enhance the abilities of structural optimization. By doing that, we divide the input space to form the premise part of the fuzzy rules and the consequence part of each fuzzy rule is newly organized based on center points of data group extracted by the HCM clustering. It concerns the fuzzy model-related parameters such as the number of input variables, a collection of specific subset of input variables, the number of membership functions, and the polynomial type of the consequence part of fuzzy rules. In the hybrid optimization process, two general optimization mechanisms are explored. The structural optimization is realized via HFCGA and HCM method whereas in case of the parametric optimization we proceed with a standard least square method as well as HFCGA method as well. A comparative analysis demonstrates that the proposed algorithm is superior to the conventional methods.

1 Introduction

There has been a diversity of approaches to fuzzy modeling. To enumerate a few representative trends, it is essential to refer to some developments that have happened over time. In the early 1980s, linguistic modeling [1] and fuzzy relation equation-based approach [2] were proposed as primordial identification methods for fuzzy models. The general class of Sugeno-Takagi models [3] gave rise to more sophisticated rule-based systems where the rules come with conclusions forming local regression models. While appealing with respect to the basic topology (a modular fuzzy model composed of a series of rules) [4], these models still await formal solutions as far as the structure optimization of the model is concerned, say a

construction of the underlying fuzzy sets—information granules being viewed as basic building blocks of any fuzzy model.

Some enhancements to the model have been proposed by Oh and Pedrycz [5]. As one of the enhanced fuzzy model, information granulation based fuzzy relation fuzzy model was introduced. Over there, binary-coded genetic algorithm was used to optimize structure and premise parameters of fuzzy model, yet the problem of finding “good” initial parameters of the fuzzy sets in the rules remains open.

This study concentrates on optimization of information granulation-oriented fuzzy model. Also, we propose to use hierarchical fair competition genetic algorithm (HFCGA) for optimization of fuzzy model. GAs is well known as an optimization algorithm which can be searched global solution. It has been shown to be very successful in many applications and in very different domains. However it may get trapped in a sub-optimal region of the search space thus becoming unable to find better quality solutions, especially for very large search space. The parallel genetic algorithm (PGA) is developed with the aid of global search and retard premature convergence [8]. In particular, as one of the PGA model, HFCGA has an effect on a problem having very large search space [9].

In the sequel, the design methodology emerges as two phases of structural optimization (based on Hard C-Means (HCM) clustering and HFCGA) and parametric identification (based on least square method (LSM), as well as HCM clustering and HFCGA). Information granulation with the aid of HCM clustering helps determine the initial parameters of fuzzy model such as the initial apexes of the membership functions and the initial values of polynomial function being used in the premise and consequence part of the fuzzy rules. And the initial parameters are adjusted effectively with the aid of the HFCGA and the LSM.

2 Design of Fuzzy Model Based on Information Granulation

Usually, information granules [6] are viewed as related collections of objects (data point, in particular) drawn together by the criteria of proximity, similarity, or functionality. Granulation of information is an inherent and omnipresent activity of human beings carried out with intent of gaining a better insight into a problem under consideration and arriving at its efficient solution. In particular, granulation of information is aimed at transforming the problem at hand into several smaller and therefore manageable tasks. In this way, we partition this problem into a series of well-defined subproblems (modules) of a far lower computational complexity than the original one. The form of information granulation (IG) themselves becomes an important design feature of the fuzzy model, which are geared toward capturing relationships between information granules.

The identification procedure for fuzzy models is usually split into the identification activities dealing with the premise and consequence parts of the rules. The identification completed at the premise level consists of two main steps. First, we select the input variables x_1, x_2, \dots, x_k of the rules. Second, we form fuzzy partitions of the spaces over which these individual variables are defined. The identification of the consequence part of the rules embraces two phases, namely 1) a selection of the consequence variables of the fuzzy rules, and 2) determination of the parameters of

the consequence (conclusion part). And the least square error (LSE) method used at the parametric optimization of the consequence parts of the successive rules.

In the premise part of the rules, we confine ourselves to a triangular type of membership functions whose parameters are subject to some optimization. The HCM clustering [7] helps us organize the data into cluster so in this way we capture the characteristics of the experimental data. In the regions where some clusters of data have been identified, we end up with some fuzzy sets that help reflect the specificity of the data set.

The identification of the premise part is completed in the following manner. Given is a set of data $\mathbf{U}=\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_l; \mathbf{y}\}$, where $\mathbf{x}_k=[x_{1k}, \dots, x_{mk}]^T$, $\mathbf{y}=[y_1, \dots, y_m]^T$, l is the number of variables and m is the number of data.

[Step 1] Arrange a set of data \mathbf{U} into data set \mathbf{X}_k composed of respective input data and output data

$$\mathbf{X}_k=[\mathbf{x}_k; \mathbf{y}] \tag{1}$$

\mathbf{X}_k is data set of k -th input data and output data, where, $\mathbf{x}_k=[x_{1k}, \dots, x_{mk}]^T$, $\mathbf{y}=[y_1, \dots, y_m]^T$, and $k=1, 2, \dots, l$.

[Step 2] Complete the HCM clustering to determine the centers (prototypes) \mathbf{v}_{kg} with data set \mathbf{X}_k .

[Step 3] Partition the corresponding isolated input space using the prototypes of the clusters \mathbf{v}_{kg} . Associate each clusters with some meaning (semantics), say Small, Big, etc.

[Step 4] Set the initial apexes of the membership functions using the prototypes \mathbf{v}_{kg} .

After premise part of identification, we identify the structure considering the initial values of the polynomial functions based on the information granules realized for the consequence and antecedents parts.

[Step 1] Find a set of data included in the fuzzy space of the j -th rule.

[Step 2] Compute the prototypes \mathbf{V}_j of the data set by taking the arithmetic mean of each rule

$$\mathbf{V}_j = \{V_{1j}, V_{2j}, \dots, V_{kj}; M_j\}. \tag{2}$$

[Step 3] Set the initial values of polynomial functions with the center vectors \mathbf{V}_j .

The identification of the conclusion parts of the rules deals with a selection of their structure (type 1, type 2, type 3 and type 4) that is followed by the determination of the respective parameters of the local functions occurring there.

The conclusion part of the rule that is extended form of a typical fuzzy rule in the TSK (Takagi-Sugeno-Kang) fuzzy model has the form

$$R^j : \text{If } x_1 \text{ is } A_{1c} \text{ and } \dots \text{ and } x_k \text{ is } A_{kc} \text{ then } y_j - M_j = f_j(x_1, \dots, x_k). \tag{3}$$

Type 1 (Simplified Inference):

$$f_j = a_{j0} \tag{4}$$

Type 2 (Linear Inference):

$$f_j = a_{j0} + a_{j1}(x_1 - V_{1j}) + \dots + a_{jk}(x_k - V_{jk}) \tag{5}$$

Type 3 (Quadratic Inference):

$$f_j = a_{j0} + a_{j1}(x_1 - V_{1j}) + \dots + a_{jk}(x_k - V_{kj}) + a_{j(k+1)}(x_1 - V_{1j})^2 + \dots + a_{j(2k)}(x_k - V_{kj})^2 + a_{j(2k+1)}(x_1 - V_{1j})(x_2 - V_{2j}) + \dots + a_{j((k+2)(k+1)/2)}(x_{k-1} - V_{(k-1)j})(x_k - V_{kj}) \tag{6}$$

Type 4 (Modified Quadratic Inference):

$$f_j = a_{j0} + a_{j1}(x_1 - V_{1j}) + \dots + a_{jk}(x_k - V_{kj}) + a_{j(k+1)}(x_1 - V_{1j})(x_2 - V_{2j}) + \dots + a_{j(k(k+1)/2)}(x_{k-1} - V_{(k-1)j})(x_k - V_{kj}) . \tag{7}$$

The calculations of the numeric output of the model, based on the activation (matching) levels of the rules there, rely on the following expression

$$y^* = \frac{\sum_{j=1}^n w_{ji}y_i}{\sum_{j=1}^n w_{ji}} = \frac{\sum_{j=1}^n w_{ji}(f_j(x_1, \dots, x_k) + M_j)}{\sum_{j=1}^n w_{ji}} = \sum_{j=1}^n \hat{w}_{ji}(f_j(x_1, \dots, x_k) + M_j) \tag{8}$$

Here, as the normalized value of w_{ji} , we use an abbreviated notation to describe an activation level of rule R^j to be in the form

$$\hat{w}_{ji} = \frac{w_{ji}}{\sum_{j=1}^n w_{ji}} , \hat{w}_{ji} = \frac{A_{j1}(x_{1i}) \times \dots \times A_{jk}(x_{ki})}{\sum_{j=1}^n A_{j1}(x_{1i}) \times \dots \times A_{jk}(x_{ki})} , \tag{9}$$

where, R^j is the j -th fuzzy rule, x_k represents the input variables, A_{kc} is a membership function of fuzzy sets, a_{jk} is a constant, V_{jk} and M_j is a center value of the input and output data, respectively, n is the number of fuzzy rules, y^* is the inferred output value, w_{ji} is the premise fitness matching R^j (activation level).

The consequence parameters a_{jk} can be determined by the standard least-squares method that leads to the expression

$$\hat{\mathbf{a}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y} \tag{10}$$

In the case of Type 2 we have

$$\hat{\mathbf{a}} = [a_{10} \dots a_{n0} \ a_{11} \dots a_{n1} \dots a_{1k} \dots a_{nk}]^T , \mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \dots \ \mathbf{x}_i \ \dots \ \mathbf{x}_m]^T ,$$

$$\mathbf{x}_i^T = [\hat{w}_{1i} \dots \hat{w}_{ni} \ (x_{1i} - V_{11})\hat{w}_{1i} \dots (x_{1i} - V_{1n})\hat{w}_{ni} \dots (x_{ki} - V_{k1})\hat{w}_{1i} \dots (x_{ki} - V_{kn})\hat{w}_{ni}] ,$$

$$\mathbf{Y} = \left[y_1 - \left(\sum_{j=1}^n M_j w_{j1} \right) \quad y_2 - \left(\sum_{j=1}^n M_j w_{j2} \right) \quad \dots \quad y_m - \left(\sum_{j=1}^n M_j w_{jm} \right) \right]^T .$$

$$\mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \dots \ \mathbf{x}_i \ \dots \ \mathbf{x}_m]^T .$$

3 HFCGA and Optimization of Fuzzy Model

One of the central problems in evolutionary computation is to combat premature convergence and to achieve balanced exploration. Parallel Genetic Algorithm (PGA) is devised to solve this problem, and there are various PGA models such as global, fine-grained, and coarse-grained model [8]. The most popular model is coarse-grained model and Hierarchical Fair Competition model (HFC) is one type of PGA. It has multiple-deme (subpopulation), individuals evolve within each deme independently, and specified individuals migrate to other deme in regular generation interval. Evolutionary process is similar to traditional GAs, but it include migration algorithm.

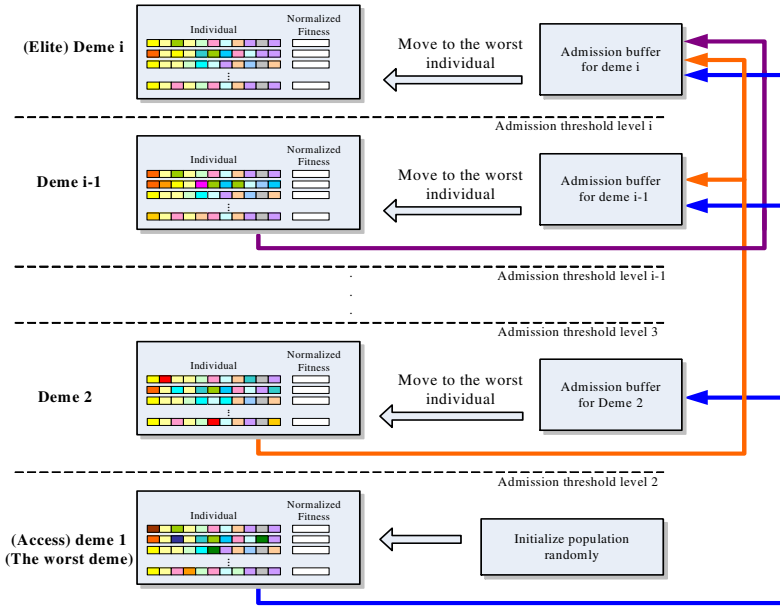


Fig. 1. The migration topology of HFCGA

In HFCGA, migration is executed in regular generation interval. And procedure is explained in detail like as follows, refer to Fig. 1.

[Step 1] Normalize the fitness of individuals in each subpopulation

$$nf_{j,i} = \frac{f_{j,i} - f_{\min}}{f_{\max} - f_{\min}}, \tag{11}$$

where, f_{ij} is fitness of i -th deme and j -th individual, f_{\max} and f_{\min} is maximum and minimum value of fitness, respectively.

[Step 2] Calculate admission threshold (A_{Li}). We use average of normalized fitness to determine admission threshold for the i 'th deme.

$$A_{Li} = \frac{1}{n_i} \sum_{j=1}^{n_i} n f_{j,i}^f, \tag{12}$$

where, n_i is a size of i -th subpopulation.

[Step 3] Create admission buffer to collect qualified candidates from other populations. All individuals with higher normalized fitness than the boundary of their deme (superior individuals) are saved to admission buffer of the appropriate subpopulation.

[Step 4] Move individuals from each admission buffer to appropriated deme. In each hierarchy deme, individuals in admission buffer exchange to the worst in deme in sequence. If normalized fitness of migrant individual is lower than the worst, pass to next individual without moving. So, there is no change if all of them in admission buffer have lower normalized fitness than the worst in target deme. And individuals in the access deme are initialized randomly to keep diversity of population.

Optimization procedure of fuzzy model consists of two phase, structural and parametric parts. In each optimization phase arrangement of chromosomes are shown as Fig. 2. Since we use real-coded HFCGA, allocate one memory space per parameter and chromosomes have real number.

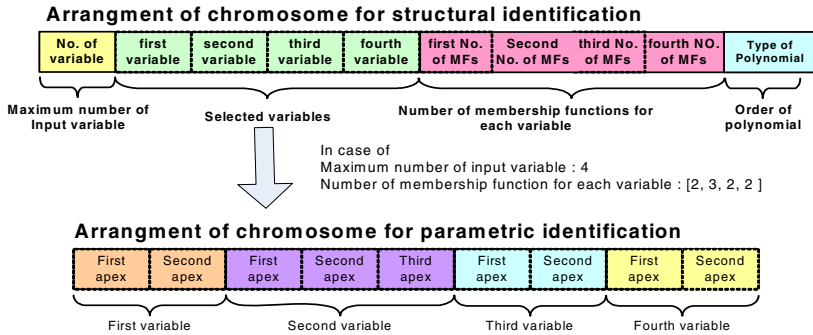


Fig. 2. Arrangement of chromosomes for identification of structure and parameters

4 Experimental Studies

We Consider a Medical Imaging System (MIS) subset of 390 software modules written in Pascal and FORTRAN for modeling. These modules consist of approximately 40,000 lines of code. To design an optimal model from the MIS, we study 11 system input variables such as, *LOC*, *CL*, *TChar*, *TComm*, *MChar*, *DChar*, \hat{N} , \hat{N} , *NF*, *V(G)* and *BW*. The output variable of the model is the number of changes *Changes* made to the software model during its development. In case of the MIS data, the performance index is defined as the mean squared error (MSE). Table 1 summarizes the list of parameters and operators used HFCGA.

Table 1. Summary of the parameters of HFCGA

| GA parameters | Structure Identification. | Parameter Identification. |
|--------------------|---------------------------|-----------------------------|
| Generation | 300 | 300 |
| No. of deme | 3 | 5 |
| Sizes of demes | [50, 50, 50] | [80, 80, 80, 80, 80] |
| Crossover rate | [0.75 0.65 0.5] | [0.75, 0.7, 0.65, 0.6, 0.5] |
| Mutation rate | [0.1, 0.1, 0.1] | [0.1, 0.1, 0.1, 0.1, 0.1] |
| Migration interval | 20 | 20 |

In structure optimization based on HFCGA, two input variables (*TComm* and *MChar*) are selected, numbers of membership functions are three and two for each selected variable, and the order of consequence polynomial is three. Table 2 show the optimized structure and performance index in HFCGA.

In the Fig. 3, upper parts depicts groups and central values through HCM for each selected input variable, where central values are used to design IG based fuzzy model and used as apexes of the membership functions in structure optimization, and lower parts represent tuned apexes of membership functions in parameter optimization.

Table 2. Performance index of IG-based fuzzy model by means of HFCGA

| Selected input variables | Structure Identification | | | Parameter Identification. | | |
|--------------------------|--------------------------|---------------------|---------|---------------------------|---------|---------|
| | No. of MFs | Order of polynomial | PI | E_PI | PI | E_PI |
| <i>TComm</i> | 3 | Type 3 | 29.6447 | 37.7859 | 29.9692 | 26.8127 |
| <i>MChar</i> | 2 | | | | | |

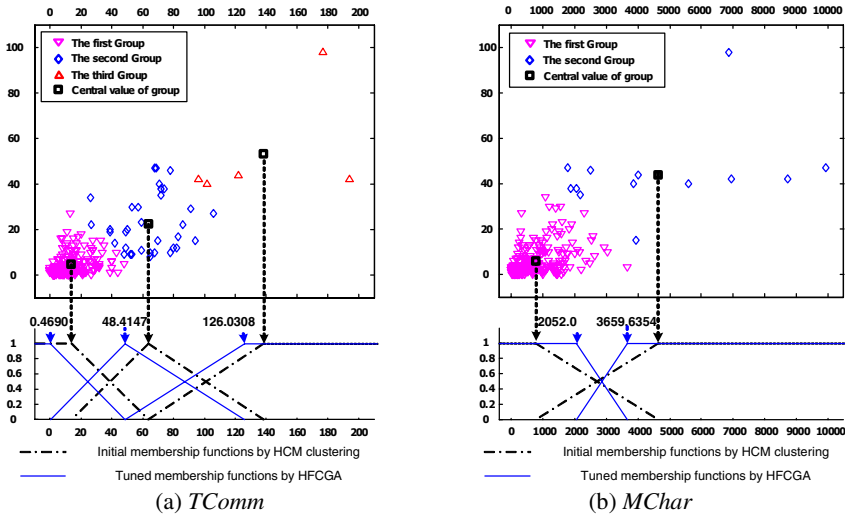


Fig. 3. Results of HCM clustering and tuned apexes of membership functions

Table 3 summarizes the results of comparative analysis of the proposed model with respect to other constructs. We use performance index of the first layer of other models for rational comparison because of the fuzzy relation model is different from FNN, SONFN and FPNN in model structure.

Table 3. Comparison of performance index with previous model

| Model | Selected inputs | No. of MFs (rules) | Polynomial Type | PI | E_PI |
|------------|-------------------------------|--------------------|-----------------|--------|--------|
| Regression | All | | | 40.056 | 36.322 |
| Model [10] | <i>TComm, MChar, DChar, N</i> | | | 43.849 | 38.917 |
| SONFN [11] | <i>TComm, MChar, DChar, N</i> | 2×2×2(16) | 2 | 39.179 | 23.864 |
| FPNN [12] | <i>TComm, N</i> | 3×3(9) | 1 | 51.005 | 36.352 |
| | <i>TChar, TComm, N</i> | 2×3×2(18) | 1 | 38.482 | 30.508 |
| Our model | <i>TComm, MChar</i> | 3×2(6) | 3 | 29.969 | 26.813 |

5 Conclusions

In this paper, we have developed a comprehensive hybrid identification framework for information granulation-oriented fuzzy model using hierarchical fair competition genetic algorithm. The underlying idea deals with an optimization of information granules by exploiting techniques of clustering and genetic algorithms. We used the isolated input space for each input variable and defined the fuzzy space by information granule. Information granulation with the aid of HCM clustering help determine the initial parameters of fuzzy model such as the initial apexes of the membership functions and the initial values of polynomial function being used in the premise and consequence part of the fuzzy rules. The initial parameters are fine-tuned (adjusted) effectively with the aid of HFCGA and the least square method. The experimental studies showed that the model is compact (realized through a small number of rules), and its performance is better than some other previous models. The proposed model is effective for nonlinear complex systems, so we can construct a well-organized model.

Acknowledgements. This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD)(KRF-2006-311-D00194).

References

1. Tong RM.: Synthesis of fuzzy models for industrial processes. *Int. J Gen Syst.* 4 (1978) 143-162
2. Pedrycz, W.: An identification algorithm in fuzzy relational system. *Fuzzy Sets Syst.* 13 (1984) 153-167
3. Takagi, T., Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans Syst, Cybern.* **SMC-15**(1) (1985) 116-132
4. Sugeno, M., Yasukawa, T.: Linguistic modeling based on numerical data. In: *IFSA'91 Brussels, Computer, Management & System Science.* (1991) 264-267

5. Oh, S.K., Pedrycz, W.: Identification of fuzzy systems by means of an auto-tuning algorithm and its application to nonlinear systems. *Fuzzy Sets and Syst.* **115**(2) (2000) 205-230
6. Pderycz, W., Vukovich, G.: Granular neural networks. *Neurocomputing.* 36 (2001) 205-224
7. Krishnaiah, P.R., Kanal, L.N., editors.: Classification, pattern recognition, and reduction of dimensionality, volume 2 of *Handbook of Statistics*. North-Holland, Amsterdam. (1982)
8. Lin, S.C., Goodman, E., Punch, W.: Coarse-Grain Parallel Genetic Algorithms: Categorization and New Approach. *IEEE Conf. on Parallel and Distrib. Processing.* Nov. (1994)
9. Hu, J.J., Goodman, E.: The Hierarchical Fair Competition (HFC) Model for Parallel Evolutionary Algorithms. *Proceedings of the 2002 Congress on Evolutionary Computation: CEC2002.* IEEE. Honolulu. Hawaii. (2002)
10. Lyu, M.R.: *Handbook of Software Reliability Engineering.* McGraw-Hill, New York. 1995 510-514
11. Oh, S.K., Pderycz, W., Park, B.J.: Self-organizing neurofuzzy networks in modeling software data. *Fuzzy Sets and Systems.* 145 (2004) 165-181
12. Oh, S.K., Lee, I.T., Choi, J.N.: Design of Fuzzy Polynomial Neural Networks with the Aid of Genetic Fuzzy Granulation and Its Application to Multi-variable Process System. *Lecture Notes in Computer Science* 3971 (2006) 774-779

Potential Assessment of an Ellipsoidal Neural Fuzzy Time Series Model for Freeway Traffic Prediction

Ping-Feng Pai¹, Kuo-Ping Lin², and Ping-Teng Chang²

¹Department of Information Management, National Chi Nan University
1 University Rd., Puli, Nantou, 545, Taiwan, R.O.C.
paipf@ncnu.edu.tw

²Department of Industrial Engineering and Enterprise Information, Tunghai University
Box 985, Taichung 407, Taiwan, R.O.C.
d923306@thu.edu.tw, ptchang@ie.thu.edu.tw

Abstract. Forecasting of traffic flow is one of the most important approaches to control the capacity of highway network efficiently during peak flow periods. Therefore, many emerging methods have been designed to predict traffic flow of freeways. However, the ellipsoidal neural fuzzy model, originally developed for control and pattern recognition problems, was seldom used in forecasting traffic flow. The aim of this study is to investigate the potential of ellipsoidal neural fuzzy model in predicting highway traffic. Monthly traffic data at Tai-Shan tollgate of a freeway in Taiwan are collected to depict the performance of forecasting models. Three other neural network models, namely back-propagation neural networks (BPNN), and radial basis function neural networks (RBFNN) and general regression neural networks (GRNN) models are used to predict the same traffic data sets. Simulation results reveal that the ellipsoidal neural fuzzy time-series (ENFTS) model is superior to the other models. Therefore, the ENFTS is a feasible and promising approach in predicting freeway traffic.

1 Introduction

It was reported that traffic congestion continues to impose frustrating delays on road users, and remains a growing trend in the near future [14]. Furthermore, such anomalous traffic congestion may significantly affect the system stability of lane traffics either in the time domain or in the space domain [17]. Therefore, accurate traffic forecasting can provide freeway traffic control centers with congestion information that may arise on highways. In the past decades, many models have been designed to forecast traffic flow. Stathopoulos and Karlaftis [20] presented a multivariate state space model to forecast traffic flow. They reported that the multivariate state space models reach more accurate forecasting results than univariate time series models. Smith et al. [18] used seasonal ARIMA models to forecast single point short-term traffic flow. It was concluded that the heuristic forecast generation method significantly outperforms the seasonal ARIMA model in terms of forecasting accuracy. A hybrid model combining the ARIMA model and the Kohonen neural network approach was developed by Voort et al. [21] to predict

short-term traffic in France. Experimental results indicated that the proposed hybrid model obtained more accurate results than the ARIMA model as well as the backpropagation neural network model. Kamarianakis and Prastacos [6] applied ARIMA models with space and time factors to forecast space-time stationary traffic flow. Simulation results showed that the ARIMA models successfully measure the impact of traffic flow changes. Due to the nonlinear mapping and fault-tolerance abilities, artificial neural network recently has become one of the most popular techniques in forecasting traffic. Kirby et al. [9] compared the performance of BPNN models with ARIMA models in forecasting traffic flow. According to experimental results, the BPNN model was inferior to the ARIMA model in the 30-minutes-ahead forecast. However, the BPNN model provided more accurate forecasting results than the ARIMA model for the two-hours-ahead forecasting. The BPNN models [4], [5] were applied to forecast traffic flow and speed of an inter-urban motorway networks in Netherlands and Italy, respectively. The noise of traffic flow caused a more difficult forecast in traffic flow than in speed. Ledoux [12] designed a two-steps neural network approach for predicting urban traffic flow in France. The proposed model conducted a one-minute-ahead forecasting policy and obtained fairly good forecasting accuracy. Dia [1] applied a time-lag recurrent neural network model to predict short-term traffic. Simulation results indicated that the presented neural network model can forecast speed data up to 15 minutes into the future with high accuracy. Yin et al. [22] employed a fuzzy-neural model, including a gate network and an expert network, to forecast traffic flow in an urban street network. Experimental results revealed that the proposed model outperforms the BPNN model in terms of forecasting accuracy.

Designed by Dickerson and Kosko [2], the ellipsoidal neural fuzzy system has been applied in recognizing gunshot bruise patterns [13], controlling smart cars [7], and filtering impulsive noises [8]. In this study, the ellipsoidal neural fuzzy system is modified for predicting freeway traffic volume. The rest of this article is organized as follows. The proposed ENFTS model is introduced in Sect. 2. Numerical examples and performance of the ENFTS model are depicted in Sect. 3. Finally, Sect. 4 draws conclusions.

2 The Ellipsoidal Neural Fuzzy Time Series Model

The ellipsoidal neural fuzzy system is an additive fuzzy system that can approximate any continuous or measurable function. An ellipsoidal fuzzy patch that trades the generality of fuzzy rule for the mathematical simplicity of quadratic forms is defined by a fuzzy rule. The size and shape of the ellipsoid indicates relations between inputs and outputs in certain regions of the state space. Two learning stages, an unsupervised learning phase stage and a supervised learning stage, are contained in the ellipsoidal neural system (Fig.1). In this study, the adaptive vector quantization (AVQ) system and the scaled conjugate gradient (SCG) training algorithm are used in the unsupervised learning stage and supervised learning stage respectively. The unsupervised learning approach is employed to select the ellipsoidal fuzzy rules quickly and roughly. Then the supervised learning is used to update the parameters of

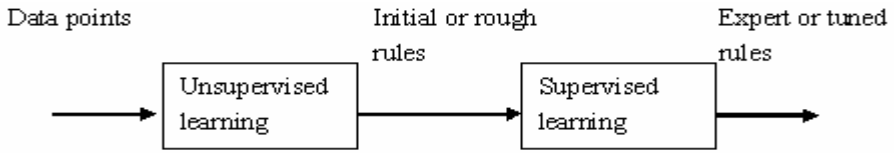


Fig. 1. The hybrid neural system combines unsupervised and supervised neural learning to find and tune the ellipsoidal fuzzy rules [3]

ellipsoidal fuzzy rules. It was reported that the hybrid system outperforms either the unsupervised learning or the supervised learning along [2].

An additive fuzzy system contains a set of rules of the form “IF X is A , then Y is B ” for fuzzy sets A and B . For a single-input and single-output function, Fig. 2 indicates the ellipsoid patches and their triangular projections. An additive fuzzy system like multilayer neural networks can approximate continuous or measurable functions uniformly by fuzzy patches. Fuzzy patches are defined by the covariance of the pattern classes in the data. In general, more fuzzy patches in shrinking sizes improve the approximation ability of an additive fuzzy system [3]. However, too many ellipsoidal fuzzy patches cause the overfitting of this fuzzy system. The influence of the ellipsoidal fuzzy patch number on the performance of the hybrid neural fuzzy system is investigated in this study.

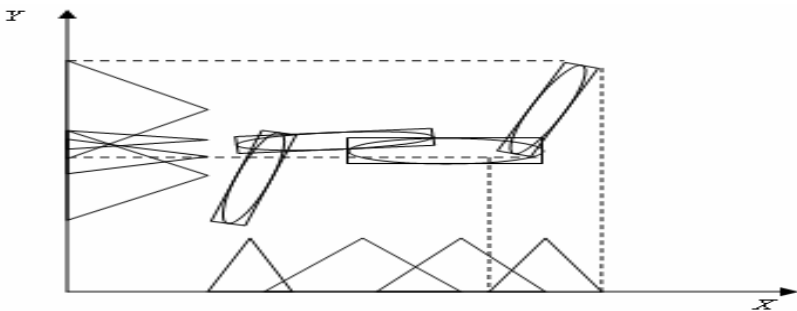


Fig. 2. The projection of each ellipsoid on the axes of the input-output state space defines a fuzzy set. The ellipsoid defines a fuzzy patch or rule between fuzzy subsets of inputs and outputs [3].

The AVQ competitive learning principle learns statistics of data clusters [10] and each data clusters develops a fuzzy ellipsoidal patch [11]. This training algorithm clusters pattern by combining the input x and the output y of the data to the form $z_j^T(t) = \left[x_j^T(t) \mid y_j^T(t) \right]$. Furthermore, the AVQ algorithm compares the vector random sample $z_j(t)$ with columns of the synaptic connection matrix $s_j(t)$. The j th neuron wins if the j th quantizing vector is expressed as in (1).

$$\|s_j(t) - z_j(t)\| = \min_i \|s_i(t) - z_i(t)\| \tag{1}$$

Competitive learning estimates the first-order statistics of the data with the stochastic difference equation showed as:

$$s_j(t+1) = \begin{cases} s_j(t) + L[z_j(t) - s_j(t)], & \text{if the } j\text{th neuron wins} \\ s_j(t) & , \text{if the } j\text{th neuron loses} \end{cases} \tag{2}$$

where L is the learning rate.

The losing vectors do not change and so do not forget what they have learned. A fuzzy ellipsoidal patch is defined by the locus of all z satisfying the following equation:

$$\alpha_j^2 = (z - c_j)^T C_j^{-1} (z - c_j) \tag{3}$$

where α_j is a positive real number, C_j^{-1} is the inverse of the covariance matrix, c_j is the center of the j th ellipsoid. The length of the projection of hyperrectangle at the j th axis onto the i th axis is shown as (4).

$$\rho_{ji} = 2 \alpha_j \sum_{k=1}^q \left(\left| \cos \gamma_{jik} \right| \right) \left(\sqrt{\lambda_{jk}} \right)^{-1} \tag{4}$$

where λ_{jk} are eigenvalues of the matrix C . The direction cosine $\cos \gamma_{jik}$ is the angle between the k th eigenvector and the i th axis for the j th ellipsoid. From the length of the projection of hyperrectangle, the triangular set can be represented as follows:

$$a_j^k(x) = \begin{cases} 1 - \frac{2|x - c_{x_{jk}}|}{\rho_{jk}}, & \text{for } |x - c_{x_{jk}}| \leq \frac{\rho_{jk}}{2} \\ 0, & \text{else} \end{cases} \tag{5}$$

In this investigation, the output at time t , O_t , is obtained by the centroidal defuzzification method with correlation product inference and showed as (6).

$$O_t = \left[\sum_{j=1}^r c_j V_j m_{A_j}(x_t) \right] \left[\sum_{j=1}^r V_j m_{A_j}(x_t) \right]^{-1} \tag{6}$$

where V_j denotes the area of the j th output set, c_j represents the centroid of the j th output set, r is the number of output sets, and $m_{A_j}(x_t)$ is the degree of fired rules.

For the supervised learning stage, this study employs a three-layer backpropagation network (Fig. 3) to train time-series data used by the ENFTS model. In the three-layer backpropagation network, the SCG algorithm is applied to adjust rules obtained from the unsupervised learning stage. Furthermore, the one-step-ahead forecasting policy is adopted. Additionally, various numbers of nodes in the hidden layers are used to study the prediction accuracy.

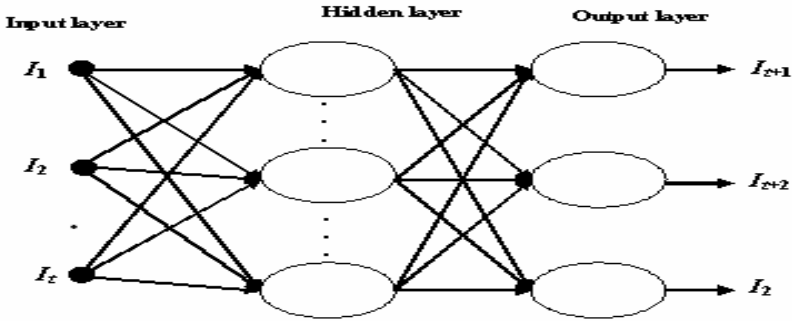


Fig. 3. The three-layer time-series neural network of the supervised phase

According to the SCG algorithm, the search direction of weights, d_{n+1} is expressed by:

$$d_{n+1} = -E'(w_{n+1}) + \beta_n d_n \tag{7}$$

Where $E(w_{n+1})$ denotes the error at the $(n+1)$ th iteration, $w_{(n+1)}$ is a vector of weights connecting two layers at iteration $(n+1)$, and β_n is represented by:

$$\beta_n = \frac{|E'(w_{n+1})|^2 - E'(w_{n+1})^T E'(w_n)}{|E'(w_n)|^2} \tag{8}$$

Thus, the weight is updated by (9).

$$\Delta w_n = \varepsilon_n d_n \tag{9}$$

The step size ε_n is shown as (10):

$$\varepsilon_n = \left[-d_n^T E'(w_n) \right] \left[d_n^T g_n + \lambda_n |d_n|^2 \right]^{-1} \tag{10}$$

where g_n and λ_n are illustrated as (11) and (12)

$$g_n = [E'(w_n + \sigma d_n) - E'(w_n)] \sigma^{-1}, 0 < \sigma < 1 \tag{11}$$

$$\lambda_n = \begin{cases} \frac{1}{2} \lambda_n, & \text{if } \Delta_n > 0.75 \\ \lambda_n, & \text{if } 0.25 \leq \Delta_n \leq 0.75 \\ 4 \lambda_n, & \text{if } \Delta_n < 0.25 \end{cases} \tag{12}$$

where

$$\Delta_n = \left\{ 2d_n^T g_n [E(w_n) - E(w_n + \varepsilon_n d_n^T)] \right\} [-d_n^T E'(w_n)]^{-2} \tag{13}$$

3 A Numerical Example

In the past, Tai-Shan tollgate had the highest traffic volume among all tollgates in Taiwan. In this investigation, monthly traffic data from Jan. 2002 to Dec. 2005 at Tai-Shan tollgate are used to demonstrate the performance of forecasting models. Fig. 4 shows the experimental data employed in this investigation. The 48 data are divided into a training data set (from Jan. 2002 to Dec. 2003) and a testing data set (from Jan. 2004 to Dec. 2005). In addition, three other neural networks, namely the back-propagation network [16], the radial basis function neural network [15], and the general regression neural network [19] are used to compare performance with the ENFTS model. Because values of parameters used by prediction models influence forecasting accuracy a lot, parameter selections are conducted for each model. For the ENFTS model, two major parameters are the rule number in the unsupervised learning stage and the number of hidden nodes in the supervised learning stage. For the BPNN model with one hidden layer, the number of hidden nodes is a main parameter to determine the prediction performance. For both the RBFNN model and the GRNN model, width of the Gaussian function (σ) is the key parameter to influence forecasting accuracy. Figure 5 shows the relation between forecasting NRMSE values and two parameters of ENFTS models. It is indicated that the smallest NRMSE value (0.02616) is obtained when the number of rules and the number of hidden nodes are 10 and 3 respectively. It is shown in Fig. 6 that the BPNN model can generate the smallest forecasting NRMSE value when the number of hidden nodes is equal to one. Figure 6 indicates the overfitting of the BPNN model when the number of hidden nodes increases. Figure 7 depicts the relation between σ values and predicting NRMSE values of RBFNN and GRNN models. The RBFNN and GRNN models can achieve the most accurate forecasting accuracy while the widths of the Gaussian functions are 4 and 2 respectively. Forecasting performances measured by NRMSE of four models are listed in Table 1. The results indicate that the ENFTS model outperforms the other three neural network models in terms of forecasting accuracy. Therefore, this may suggest ENFTS is a valid and promising model in predicting freeway traffic.

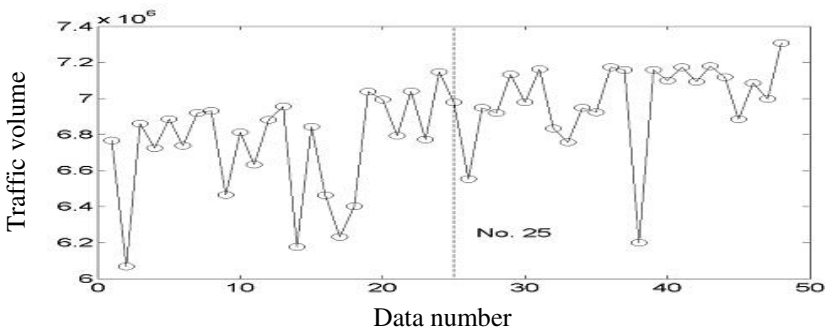


Fig. 4. The traffic volume at Tai-Shan tollgate

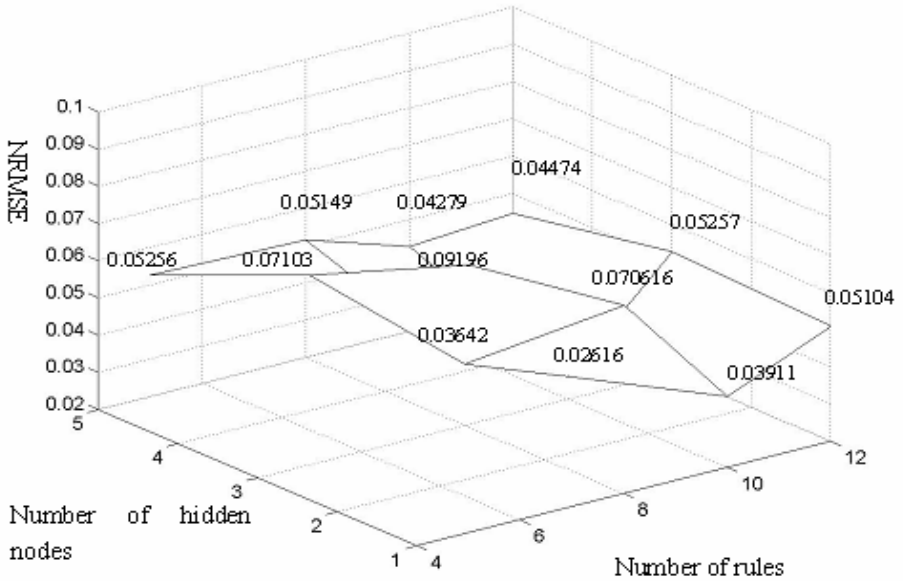


Fig. 5. The relation between NRMSE value and two parameters of ENFTS models

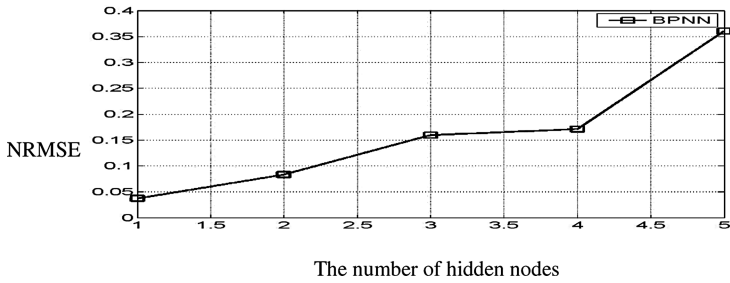


Fig. 6. The relation between the number of hidden nodes and NRMSE values of BPNN

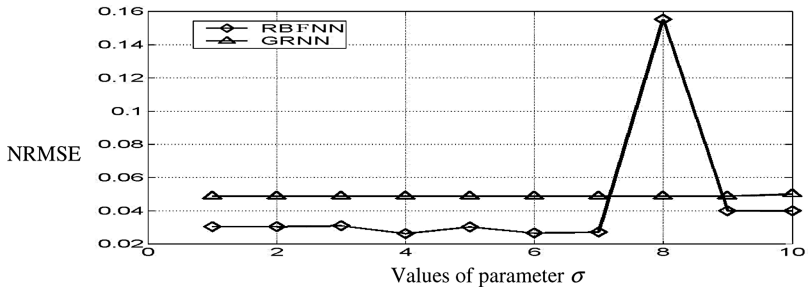


Fig. 7. The relation between σ values and NRMSE values of RBFNN and GRNN

Table 1. Forecasting accuracy of four models

| Forecasting models | ENFTS | BPNN | RBFNN | GRNN |
|--------------------|---------|---------|---------|---------|
| NRMSE | 0.02615 | 0.03726 | 0.02862 | 0.04882 |

4 Conclusions

To provide traffic control centers with useful information, accurate forecast of highway traffic is essential. This study modifies the ellipsoidal neural fuzzy model to accept time series data and applies the ENFTS model to predict freeway traffic. Numerical data collected from the busiest tollgate in Taiwan are employed to depict the forecasting performance. Forecasting results obtained by other three neural network models are used to compare the forecasting performance with the presented ENFTS model. Furthermore, parameter analyses of forecasting models are also conducted to illustrate influences of parameters on different prediction models. It is indicated that the ENFTS model is superior to the other models in forecasting accuracy. Therefore, the presented ENFTS model is feasible and promising alternative in freeway traffic prediction. For the future research, some meta-heuristic algorithms can be applied to optimizing the parameter selection of the ENFTS model.

Acknowledgements. This research was conducted with the support of National Science Council (NSC 95-2221-E-260-007).

References

1. Dia, H.: An Object-oriented Neural Network Approach to Short-term Traffic Forecasting. *European Journal of Operational Research* 131 (2001) 253–261
2. Dickerson J., Kosko B.: Fuzzy Function Learning with Covariance Ellipsoids. *Proceedings of IEEE International Conference on Fuzzy Systems, San Francisco* (1993) 1162–1167
3. Dickerson J., Kosko, B.: Fuzzy Function Approximation with Ellipsoidal Rules. *IEEE Transactions on System, Man, and Cybernetics-Part B Cybernetics* 26 (1996) 542–560
4. Dougherty M.S., Cobbett, M.R.: Short-term Inter-urban Traffic Forecasts Using Neural Networks. *International Journal of Forecasting* 13 (1997) 21–31
5. Florio, L., Mussone, L.: Neural Network Models for Classification and Forecasting of Freeway Traffic Flow Stability. *Control Engineering Practice* 4 (1996) 153–164
6. Kamarianakis, Y., Prastacos P.: Space-time Modeling of Traffic Flow. *Computers & Geosciences* 31 (2005) 119–133
7. Kim, H.M., Dickerson, J., Kosko, B.: Fuzzy Throttle and Brake Control for Platoons of Smart Cars. *Fuzzy Sets and Systems* 84 (1996) 209–234
8. Kim, H.M., Kosko B.: Fuzzy Prediction and Filtering in Impulsive Noise. *Fuzzy Sets and Systems* 77 (1996) 15–33
9. Kirby, H.R., Watson, S.M., Dougherty, M.S.: Should We Use Neural Networks or Statistical Models for Short-term Motorway Traffic Forecasting. *International Journal of Forecasting* 13 (1997) 43–50

10. Kohonen, T.: *Self-Organization and Associative Memory*. 2nd edn. Springer-Verlag, Berlin Heidelberg New York (1988)
11. Kosko, B.: *Neural Networks and Fuzzy Systems*. Prentice Hall, Englewood Cliffs New Jersey (1991)
12. Ledoux, C.: An Urban Traffic Flow Model Integrating Neural Network. *Transportation Research Part C* 5 (1997) 287–300
13. Lee, I., Kosko, B., Anderson W.F.: Modeling Gunshot Bruises in Soft Body Armor with An Adaptive Fuzzy System. *IEEE Transactions on System, Man, and Cybernetics-Part B Cybernetics* 35 (2004) 1374–1390
14. Lindley, J.A.: Urban Freeway Congestion: Quantification of the Problem and Effectiveness of Potential Solutions. *Institute of Transportation Engineers Journal* 57 (1987) 27–32
15. Moody, J., Darken, C.: Fast Learning in Networks of Locally-tuned Processing Units. *Neural Compute* 1 (1989) 281–94
16. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning Internal Representations by Back-propagating Errors. In: Rumelhart, D.E., McClelland, J.L., The PDP Research Group (Eds.): *Parallel Distributed Processing*, Vol. 1, MIT Press, Cambridge (1986) 318–362
17. Sheu, J.-B., Chou, Y.-H., Shen, L.-J.: A Stochastic Estimation Approach to Real-time Prediction of Incident Effects on Freeway Traffic Congestion. *Transportation Research Part B* 35 (2001) 575–592
18. Smith, B.L., Williams, B.M., Oswald, R.K.: Comparison of Parametric and Nonparametric Models for Traffic Flow Forecasting. *Transportation Research Part C* 10 (2002) 303–321
19. Specht, D.F.: A General Regression Neural Network. *IEEE Transactions on Neural Networks* 2 (1991) 568–576
20. Stathopoulos, A., Karlaftis, G.M.: A Multivariate State Space Approach for Urban Traffic Flow Modeling and Prediction. *Transportation Research Part C* 11 (2003) 121–135
21. Voort, Maschavan Van Der, M., Dougherty, M., Watson, S.: Combining Kohonen Maps with ARIMA Time Series Models to Forecast Traffic Flow. *Transportation Research Part C* 4 (1996) 307–318
22. Yin, H., Wong, S.C., Xu, J., Wong, C.K.: Urban Traffic Flow Prediction Using a Fuzzy-neural Approach. *Transportation Research Part C* 10 (2002) 85–98

Digital Model of Series Resonant Converter with Piezoelectric Ceramic Transducers and Fuzzy Logic Control

Paweł Fabijański and Ryszard Łagoda

Institute of Control and Industrial Electronics, Warsaw University of Technology,
Koszykowa 75, 00-662 Warszawa, Poland
pawel@isep.pw.edu.pl, lagoda@isep.pw.edu.pl

Abstract Sandwich type piezoelectric ceramic transducers are the most frequently applied source of ultrasounds in technical cleaning system. They have the ability to radiate in an ultrasonic medium, e.g. water, with maximum acoustic power when the vibration is activated by a current whose frequency equals the mechanical resonance frequency of the transducer. In resonant inverters the transducer units are part of the oscillating circuit, for which equivalent electrical circuit consist of connection in parallel: Co end RLC. The resonant frequency of the real circuit varies during the operation in function of many parameters, among others, the most important are temperature, time, the column of cleaning factor, and the surface of the cleaned elements. In this situation, to obtain the maximum value of the converter efficiency, its important role of control system to assure the optimal mechanical resonant frequencies of converter.

1 Introduction

High power ultrasonic waves are generally used in such industrial processes as welding, acceleration of chemical reactions, scavenging in gas medium, echo sounding and underwater communication (sonar systems), picture transmission, and, above all, ultrasonic cleaning.

Currently, the Sandwich-type piezoelectric ceramic transducers are the most frequently applied sources of ultrasound. They have the ability to radiate in an ultrasonic medium with maximum acoustic power when the vibration is activated by a current whose frequency equals the mechanical resonance frequency of the transducer. Typical units of the ultrasonic generators feeding these transducers operate at frequencies between 20 kHz and 100 kHz (Fig. 1), with output power in the range of 20 W to 5 kW.

In resonant converters, the SANDWICH-type transducer units is a part of the oscillating circuit. This transducer made of piezoelectric ceramics PZT and are a combination of steel-ceramics-aluminum blocks connected by one or several screws.

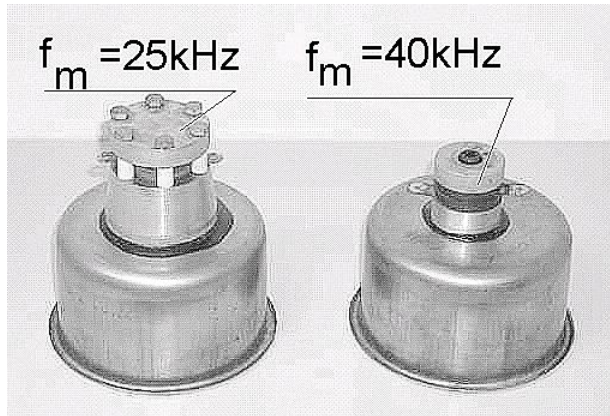


Fig. 1. Ultrasonic SANDWICH-type transducers

2 Power Circuit Configuration

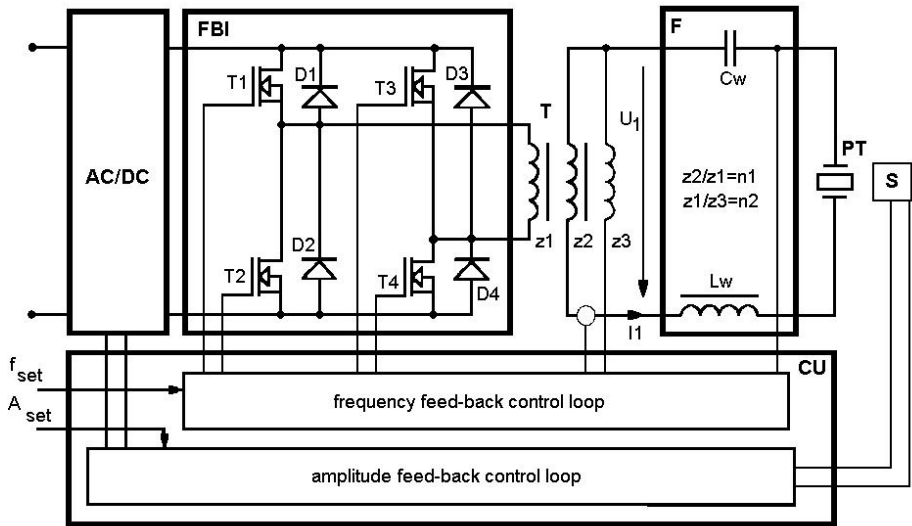


Fig. 2. The block diagram of the ultrasonic generator

Block diagram and the main circuit of the converter with piezoelectric ceramic transducer (Fig. 2) consists of:

- converter AC/DC,
- full-bridge inverter (T1-T4, D1-D4) FBI,
- isolating transformer T, where $z_2/z_1=n_1$, $z_1/z_3=n_2$,
- special filter (Lw, Cw) F,

- transducer PT,
- sensor of vibrations S,
- control unit CU.

The control unit consists of two parts. The first part, in FBI inverter, is the frequency feed-back control loop and the second part in AC/DC converter is the amplitude feed-back control loop.

Two coupling loop works independently.

In the case of overload inverter system of current limiter disconnect supply of AC/DC converter.

Signal f_{set} make possible to set up manually frequency switching inverter FBI and signal A_{set} establish amplitude ultrasonic oscillation.

3 Digital Model of Inverter-Special Filter-Transducer Group System Circuit

The equivalent circuit of the piezoelectric ceramic transducers with frequency close to resonant frequency is shown in Figure 3, where:

C_0 - static capacity of the transducer,

C - equivalent mechanical capacity,

L - equivalent mechanical inductance,

R - equivalent resistance, $R_p = R_m + R_a$,

where: R_m - equivalent mechanical loss resistance, R_a - equivalent acoustic resistance.

The values C_0 , C , L , R are calculated from admittance characteristic of transducers (Fig. 4), for the off-load and on-load conditions ($R_a = \infty$).

Exemplary values of these parameters for the transducer immersed in air may be following: $C_0 = 4 \text{ nF}$, $L = 246 \text{ mH}$, $C = 182 \text{ pF}$, $R = 392 \text{ } \Omega$.

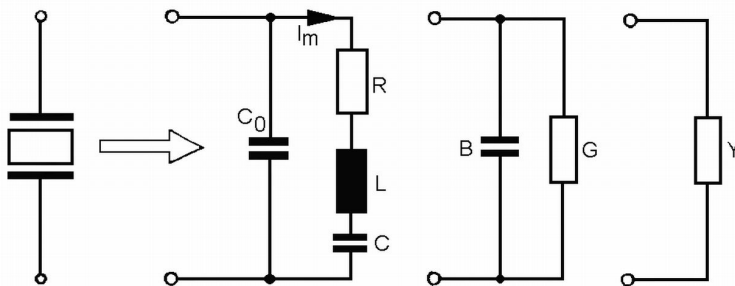


Fig. 3. The equivalent circuit of piezoelectric transducer, where : C_0 – the electrical part of circuit, RLC – the mechanical part of circuit

The susceptance B , conductance G , admittance Y of the transducers in frequency f function may be expressed by the following equations:

$$B(\omega) = \text{Im}Y(\omega) = \omega C_0 - \frac{\omega^5 LC^4 - \omega^3 C^3}{\omega^4 C^4 R^2 + (\omega^2 LC - 1)^2} \tag{1}$$

$$G(\omega) = \text{Re}Y(\omega) = \frac{\omega^2 C^2 R}{\omega^2 C^2 R^2 + (\omega^2 LC - 1)^2} \tag{2}$$

$$Y(\omega) = \sqrt{G^2 + B^2} \tag{3}$$

where $\omega = 2\pi f$.

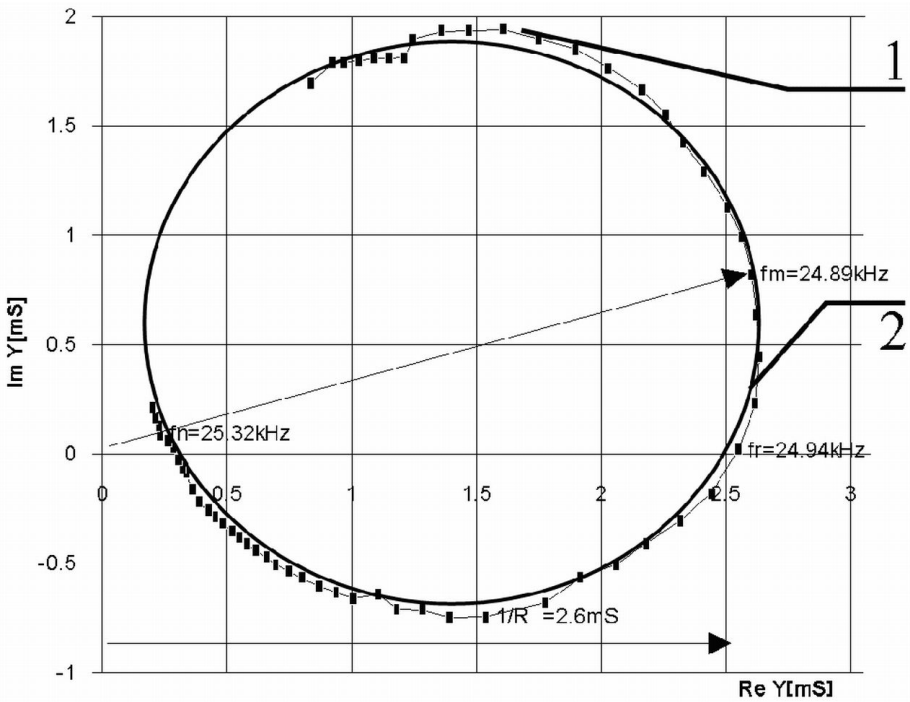


Fig. 4. The admittance characteristics of sandwich-type transducers: 1. Real characteristic, 2. Equivalent characteristic

The frequency f_m of mechanical vibration may be calculated by equation:

$$f_m = \frac{1}{2\pi} \sqrt{LC} \tag{4}$$

An optimum value inductance of choke L_w of special filter may be expressed by the equation:

$$Lw = \frac{4\pi^2 fm^2}{Co} \tag{5}$$

and value of capacity Cw is equals $Cw = Co \cdot n_2$.

The presented system of filter-transducers make possible to supply the piezoelectric ceramic transducer with the quasi-sinusoidal current and voltage and to determine a extreme values of parameters the elements of the converters FBI in case of tuning and untuning resonance frequency of converter and piezoelectric ceramic transducer ($f \approx fm$).

The digital model of the inverter-special filter- transducer group system circuit, worked in PSpice language, has enable us to analysis the current and voltage waveforms in the inverter to determine the optimum parameters for the semiconductors, and especially for analysis in the case tuning and untuning of output frequency of inverter and the mechanical frequency of the transducer. Exemplary results of digital analysis for the power circuit with open the frequency feed-back control loop are presented in Fig. 5.

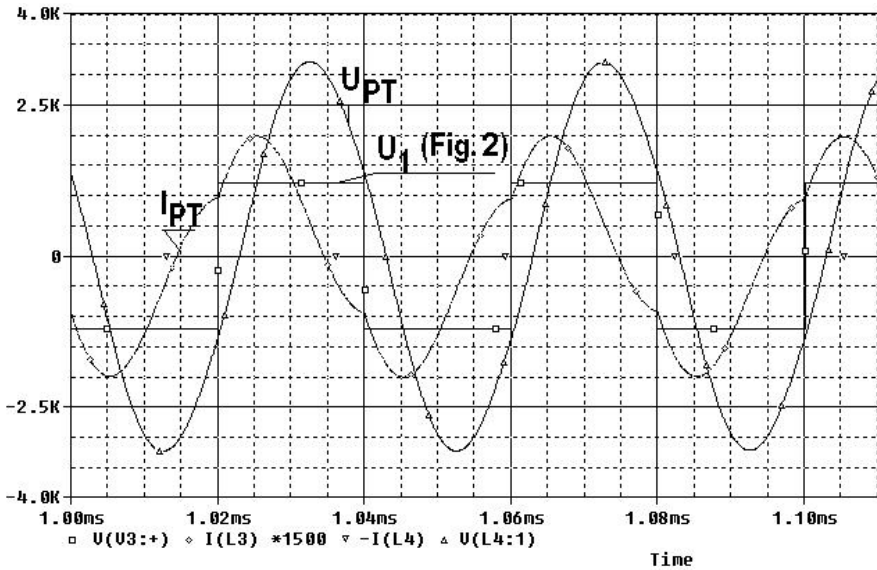


Fig. 5. Current and voltage waveform for transducer and output voltage for inverter for digital model of inverter-special filter-transducer group system circuit for $f \approx fm$

4 Control System

To obtain the maximum value of converter efficiency is necessary to assure the optimal mechanical resonant frequencies of converter, but, most of properties of piezoelectric ceramic transducer change gradually with time, depends also from the ceramic composition and the way the ceramic is processed during manufacture.

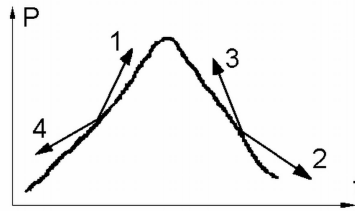


Fig. 6. Four case variations of change output power and resonant frequency

Changing the cleaning medium results in the variations of output power of our converter and in output resonant frequency we have four case (Fig. 6). In this situation, to obtain the maximum value of converter efficiency, its important role of fuzzy logic control system to assure the optimal mechanical resonant frequencies of converter.

In this situation logic control system define derived mark of signal amplitude proportional to output power and output resonant frequency and change adequate the output frequency of converter. the control rules, contain the relationship between the input and output variable are defined (Tab. 1.)

Table 1. Control rules

| | derived of power | derived of frequency | rules of logic control |
|---|------------------|----------------------|------------------------|
| 1 | $dP/dt > 0$ | $df/dt > 0$ | frequency increase |
| 2 | $dP/dt < 0$ | $df/dt > 0$ | frequency decrease |
| 3 | $dP/dt > 0$ | $df/dt < 0$ | frequency decrease |
| 4 | $dP/dt < 0$ | $df/dt < 0$ | frequency increase |

Fuzzification is process, where non-fuzzy values of power and frequency are converted into fuzzy values.

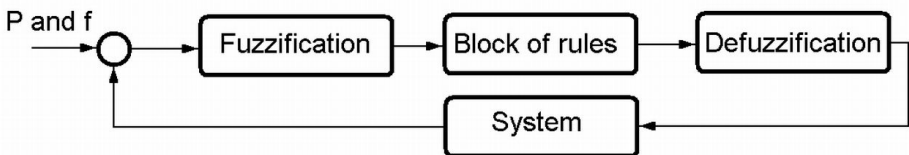


Fig. 7. Structure of Fuzzy Controller

It is done by membership function. In our case membership function have a trapezoidal shape. The most important part of fuzzy controller in a block of rules (Fig. 7). The rules were derived from system behavior. This block makes connection between input side and output side. The rules were derived for power and frequency values as:

P/dP

| | | | | | |
|----|----|----|----|----|----|
| | NB | NS | ZE | PS | PB |
| NB | NB | NB | NB | NS | NS |
| NS | NB | NB | NS | ZE | PS |
| ZE | NB | NS | ZE | PS | PB |
| PS | NS | ZE | PS | PB | PB |
| PB | PS | PS | PB | PB | PB |

With the referred lookup table a fuzzy controller was synthesized by following the control algorithm:

- Step 1: Determine the power error,
- Step 2: Determine the power error variation,
- Step 3: Quantization of the power error,
- Step 4: Quantization of the power error variation,
- Step 5: Calculate the controller’s output scale,
- Step 6: If the quantization error and error variation lever are not the smallest possible go to,
- Step 7: Adjust the universe of discourse,
- Step 8: Go to Step 3,
- Step 9: Extract the controller’s output value from the lookup table,
- Step 10: Adjust the controller’s output value using the output scale value.

Defuzzification means transfer of fuzzy value power and frequency in to non - fuzzy values.

5 Experimental Research

The main circuit of the series-resonant converter with piezoelectric ceramic transducer and special LC filter system (Fig. 2) was modeling, build and tested. The experimental result was obtain in the case of tuning and untuning of output frequency of inverter and the mechanical frequency of the transducer (circuit LC). Exemplary experimental current and voltage waveform of transducer are shown in the followed (Fig. 8 - 10).

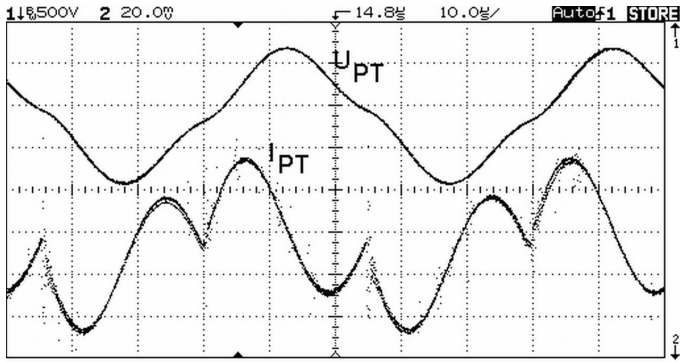


Fig. 8. Current and voltage waveform in case when $f < f_m$

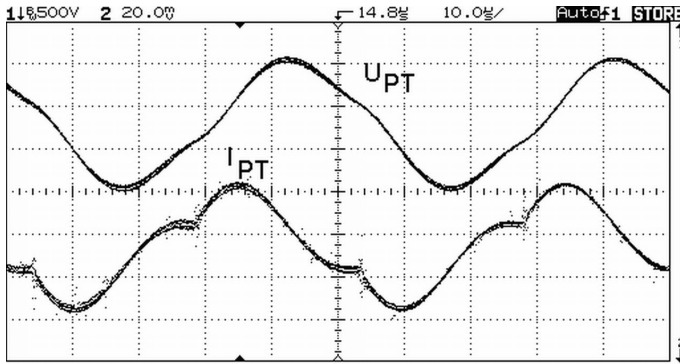


Fig. 9. Current and voltage waveform in the case when $f = f_m$

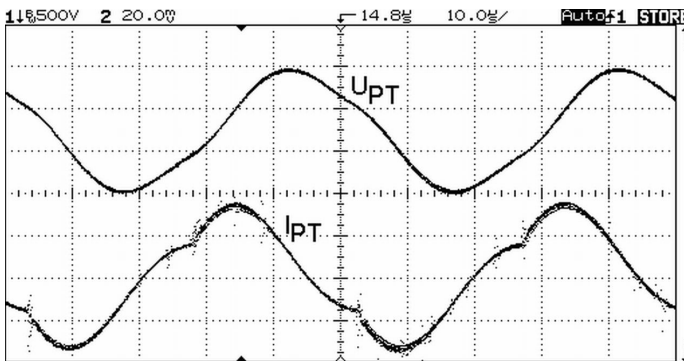


Fig. 10. Current and voltage waveform in the case when $f > f_m$

6 Conclusions

Possible application of fuzzy logic may be classified in three categories; the first one concerns problems about inaccurate and eventually uncertain knowledge, the second one concerns problem which can not be modeled rigorously, at last, problems related to very complicated systems, that is systems with a lot of variables. In this situation fuzzy logic control system define derived mark of signal amplitude proportional to output power and output resonant frequency and change adequate the output frequency of converter.

The presented control method make possible to supply the piezoelectric ceramic transducers group with the quasi-sinusoidal current and voltage with self-tuning frequency to mechanical resonance. The purpose of the analysis was to tested the control system in the case of tuning the generator frequency to the mechanic change the resonance frequency of transducer, to determine optimal parameters of the elements of resonant inverter circuit, and as well as to determine the extreme values of currents and voltages when the untuning the transducer according to the resonant frequency occurs.

In the case of untuning the output frequency of inverter according to piezoelectric ceramic transducer mechanical frequency the current and voltage waveforms of piezoelectric transducer are non-sinusoidal. Especially current waveform is deformation. Increase the o frequency make decrease current and voltage amplitude of transistors generator.

Because in the case even small untuning resonance frequency in our system the reached efficiency is deeply decrease there is necessary to used described control system which assure self tuning output voltage inverter frequency to obtain maximum output power.

The results of analysis of piezoelectric transducer and of the system of the resonance converter with control loop of frequency have been compared with experimental results in real piezoelectric transducer system and satisfactory results has been obtained.

References

1. Fabijański, P., Łagoda, R.: Control and Application of Series Resonant Converter in Technical Cleaning System. Proceedings of the IASTED, International Conference Control and Application, Cancun, Mexico, (2002)
2. Hatanaka, Y., Nakaoka, M., Maruhashi T.: Overlapping Commutation-mode Analysis of a High-frequency Inverter. International Journal Electronics, Vol. 49, No. 3, (1980)
3. Faa-Jeng, L., Rong-Jong, W., Kuo-Kai, S, Tsih-Ming, L, Recurrent Fuzzy Neural Network Control for Piezoelectric Ceramic Linear Ultrasonic Motor Drive. IEEE Transactions on Ultrasonic, Ferroelectrics, and Frequency Control, Vol. 48, No.4, (2001)

A Method to Classify Collaboration in CSCL Systems

Rafael Duque and Crescencio Bravo

Department of Information Systems and Technologies
School of Computer Engineering
University of Castilla – La Mancha
Paseo de la Universidad 4, 13071 Ciudad Real, Spain
{Rafael.Duque,Crescencio.Bravo}@uclm.es

Abstract. One of the most important challenges of collaborative learning systems is to offer mechanisms to facilitate the study of the relationships between the collaboration process and the characteristics of the solution (product) built by the learners in this work process. In this article, a machine learning algorithm that generates a set of rules to classify the different forms of collaboration within a group of learners with respect to the quality of the solution built is presented. The algorithm, based on a fuzzy model, is put into practice using data registered in a collaborative learning environment.

1 Introduction

In the last years, a growing use of information technologies is perceived in educational environments. Among the reasons that originate this situation, we can point out the adoption of some didactic theories that recommend group work [16] and the proliferation of diverse technologies to facilitate support for collective work (the Internet, wireless technologies, etc.). Consequently, new paradigms such as CSCL (Computer-Supported Collaborative Learning) [11] and ITS (Intelligent Tutoring Systems) [19] have arisen. CSCL studies the form in which technology can give support to group learning. The ITS approach seeks to introduce “intelligent” virtual teachers into teaching-learning processes.

Many CSCL systems offer analysis functions to study the way in which collaboration takes place [10]. For this purpose, they record the actions carried out with the user graphic interface [7], the conversational acts between collaborators [1], and/or the changes carried out in the shared work spaces [15]. Starting from these data, some CSCL systems calculate analysis indicators [6] using Artificial Intelligence (AI) techniques to evaluate and represent the collaboration. Such techniques include Hidden Markov Models [17], Decision Trees [5], Petri Nets [12], Plan Recognition [13], and Fuzzy Logic [2, 14]. However, in many of the cases these techniques have not been used following a machine learning approach.

In the ITS area, a great number of systems [18], based on knowledge models, evaluate the solutions created by the students. However, these systems usually lack facilities to analyze the process of solution building and they do not include support for group work. CSCL systems usually offer analysis indicators about the group work. On the contrary, ITS are focused on the quality and other characteristics of the

solution. In this work, the classification of the different forms of collaboration according to the quality of the solution built in a collaboration process is approached.

The AI techniques to be used in collaboration analysis depend largely on the model of the collaboration process and on the objective of the analysis. According to the study of Jermann et al. [10], the CSCL systems are classified in three categories: those that just reflect the interaction actions, those that monitor the state of the interaction, and those that offer advice to the users. However, most of the CSCL systems do not advise the user, in part because the AI techniques used, which are not based on fuzzy logic, do not allow the system to immediately elaborate a message in natural language close to the user since they are focused on obtaining the results by means of mathematical models (e.g., Hidden Markov Models) and not by means of a set of rules easily understandable. On the other hand, the CSCL systems that do offer advice to the users and are not based on machine learning techniques cannot build the advice according to the specific situations that take place in real-time but according to some parameters defined previously by an expert.

We hypothesised the validity of using machine learning techniques to provide fuzzy models to study how the collaboration and the work process in group interrelate with the final product (solution). This article proposes an algorithm to classify the quality of collaborative work according to the quality of the solution built in this process. This algorithm takes a set of analysis indicator values as input, referred to both collaboration and solution, and generates a set of classification rules as output.

In Section 2 the characteristics of the fuzzy models to support collaboration and solution analysis are discussed. Section 3 proposes the machine learning algorithm mentioned above. In Section 4 the application of the algorithm to some data obtained in some collaborative learning activities where several students solved design problems working in groups are presented. Finally, Section 5 shows the conclusions obtained and some future work lines.

2 Fuzzy Models to Analyze Collaborative Learning

As mentioned before, diverse uncertainty treatment techniques have been applied to analyze collaboration in CSCL systems. We propose the use of fuzzy sets [20] because they are very useful to treat the uncertainty of some collaboration analysis indicators [6] to describe concepts about the collaboration such as *communication*, *cooperation*, *agreement*, or *participation*. These concepts are difficult to define with precision but they can be described with the help of linguistic labels. In the case of CSCL systems, it could be interesting to determine, for example, the quality of the *communication* between the members of a work group. However, it is complex to define this concept with precision. For that reason, the fuzzy sets (e.g., very fluid communication, not very fluid communication, nonexistent communication, etc.) are useful to characterize the *communication* concept.

Keeping in mind that the academic subject matters are qualified with a general grade (usually a number or letter) and that CSCL systems usually offer a set of collaboration analysis indicators, a MISO (Multiple Inputs, Single Output) system has been chosen to provide a model to relate collaboration analysis indicators with the

solution quality. The inputs represent the collaborative work, and the output is a global qualification of the work made.

The rule-based fuzzy models offer easily comprehensible results. Additionally, they are easily expressible in natural language, which allows teachers without knowledge on techniques to tackle with uncertainty to manipulate rules and to receive analysis and evaluation information. In addition, the simplicity of calculating the rules allows synchronous collaboration systems to process these rules in real time.

In order to build our fuzzy model, a set of cases $\{C_0, C_1, \dots, C_m\}$ is considered. Each case C_i is made up of $k+1$ values $(x_{i0}, x_{i1}, \dots, x_{ik})$ for the corresponding analysis indicators of group work, and of a value for the solution quality (y_j) . These cases would have been calculated by the collaborative system using the users' action log and some collaboration analysis functions. Thus, the fuzzy rules have the following structure:

$$R_j: \text{If } X_0 \text{ is } L_0^i, X_1 \text{ is } L_1^i, \dots, X_k \text{ is } L_k^i, \text{ then } Y \text{ is } y_j, j=1..s$$

In brief, $R_j: \text{If } E_j \text{ then } y_j$

where (X_0, \dots, X_k) is the set of input variables (analysis indicators of collaboration), (L_0^i, \dots, L_k^i) is the fuzzy set in which the corresponding variable X takes value, and Y is the output variable that indicates the quality of the solution.

3 Machine Learning Method

The ITS and CSCL systems are used in diverse environments and learning matters. This entails that those who play the role of administrator or teacher are not familiarized with the ways of expressing uncertainty. Therefore, the aim of the classification method is to provide the smallest number of rules and with a simpler expression, and to manage variables that are independent of the application domain. A machine learning system that is completely automatic facilitates this aim since it works without the help of experts. The method proposes an algorithm to build a fuzzy model made up by a set of rules. In classification processes, a variable, data or criteria used to group or identify a set of elements is usually named *class*. In this article the *class* concept refers to a global indicator of the solution quality used to group rules. The algorithm is described below.

1. Use 80% of cases to derive initials rules
2. For $j=1$ to number of classes
 - 2.1. Group in sets those initial rules of the j class that have the same values for one or more variables from the antecedent
 - 2.2 Select the set with the greatest number of rules and find one or several rules that define the set by means of an amplification process
 - 2.3 If there are initial rules of the j class that have not yet been amplified, repeat step 2.1 but only with these initial rules

Let us analyze each one of the steps proposed in the machine learning algorithm. In the first step (step 1) a set of rules is obtained starting from the available cases. Only 80% of the cases are used, with the aim that 20% remaining can be used to check the validity of the obtained rules. If there is a case $((x_{i0}, x_{i1}, \dots, x_{ik}), y_j)$, it will be

necessary to use a mechanism that transforms the case into a rule of the type $(L_0^i, \dots, L_k^i, y_j)$. This procedure is as follows: in the first place, the value domain of each variable is divided into different intervals, each one corresponding to a linguistic label; then, the degree of membership of a value to a linguistic label is calculated. The label of the highest membership degree is assigned to the fuzzy variable.

In step 2.1., some sets of rules of the same class, that is to say, with common linguistic labels for some variables in the antecedent, are therefore created. This way, the hypothesis (to be validated afterwards) is that having common elements it is more feasible for the different rules to be defined by one or more rules that include all of them.

Among the sets obtained, the algorithm selects the one that has the greatest number of rules. The amplification process (step 2.2) consists of adding linguistic labels to the antecedent. A rule $R_i (R_i: \text{If } E_i \text{ then } y_j)$ will be amplified into another rule $R_j (R_j: \text{If } E_j \text{ then } y_j)$ where $E_i \subset E_j$, whenever a rule $R_i (R_i: \text{If } E_i \text{ then } y_i)$ does not exist and that $E_i \subset E_j$ and $y_i \neq y_j$. The algorithm follows the amplification process proposed in [4], until one or several rules that define the whole rule set are found.

Once a set of rules is obtained, its suitability is tested using those cases that were not used in the machine learning process (20%). A rule $R_i (R_i: \text{If } E_i \text{ then } y_i)$ will fulfil another rule $R_j (R_j: \text{If } E_j \text{ then } y_j)$ when $E_i \subset E_j$ and $y_i = y_j$.

4 The Method in Action

The DomoSim-TPC system [3] is a CSCL environment with support for problem solving in the Domotics domain. The tool incorporated in this system for the collaboration and solution analysis can be seen in Fig. 1. The user has to select which is the analysis target (the collaborative work, the final solution, or a collaboration-solution comprehensive study) and the type of variables to visualize (calculated, subjective, or inferred). In the right part of the user interface, the values of the selected analysis indicators are shown in a graphic way.

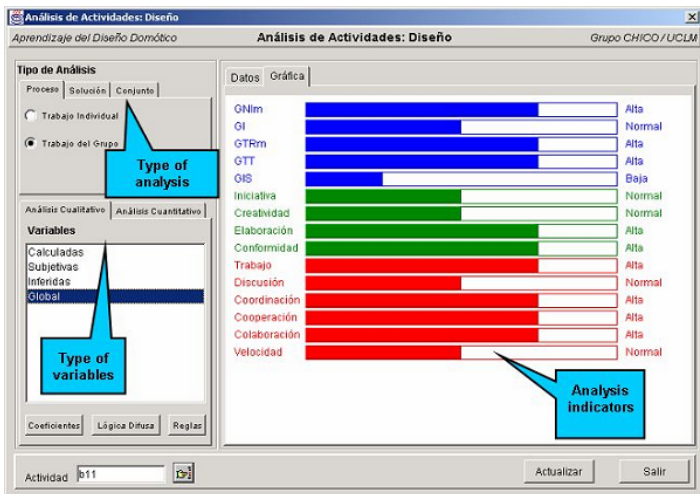


Fig. 1. Analysis tool of DomoSim-TPC

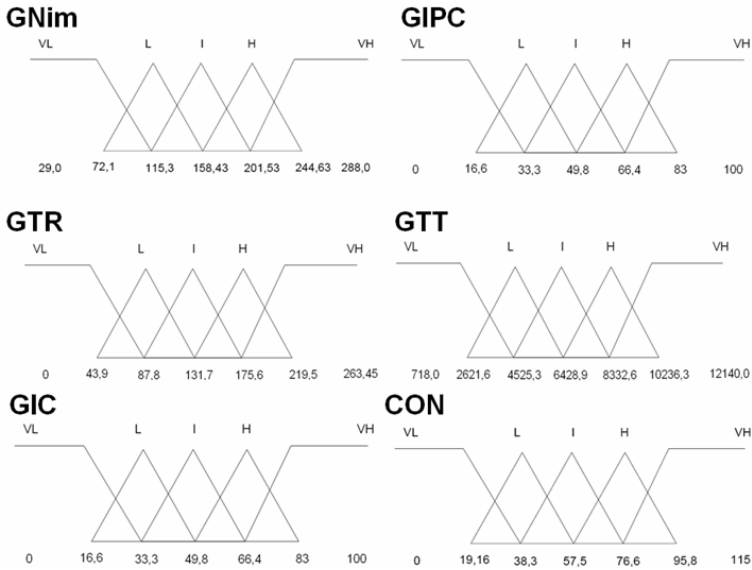


Fig. 2. Membership functions used

The DomoSim-TPC system was used in a course by secondary education students. From its usage, a data repository was made with analysis indicators of the collaborative work and a general qualification of each solution. In order to create the fuzzy model outlined above, the following analysis indicators of the collective work (calculated by DomoSim-TPC) were considered:

- GNim: Average number of actions carried out by the users of the group.
- GIPC: Interactivity of the group, calculated as the percentage of proposition and voting actions that were replied to by a user different from his/her author.
- GTR: Average time in replying to proposals or voting interactions.
- GTT: Duration of the realization of the task.
- GIC: Simulation interactivity, calculated as the percentage of simulation actions that are carried out by the users as opposed to the total number of simulation actions.
- Conformity (CON): It represents the number of actions that are carried out with a total consensus by the members of the work group.

The analysis indicator that classifies each case is the solution quality (SQUAL). This indicator measures the degree of fulfilment of the objectives defined by the problem, which implies that the solution is correct or incorrect. SQUAL can take five different values (VL: Very Low; L: Low; I: Intermediate; H: High; VH: Very High).

Once the real cases are available, it is necessary to carry out a *fuzzification* process in which the real values become linguistic labels. Concretely, the domain of each variable is divided in five intervals (VL, L, I, H, VH).

In this application of the method, five functions ($\delta_{VL}^n(x_{in})$, $\gamma_L^n(x_{in})$, $\gamma_I^n(x_{in})$, $\gamma_H^n(x_{in})$, $\mu_{VH}^n(x_{in})$) have been used for each variable X_n ($n=0..k$) as can be observed in Fig.2.

For each value x_{in} from each case, its associated linguistic label L_n^i is obtained according to the expression $\max\{\delta_{VL}^n(x_{in}), \gamma_L^n(x_{in}), \gamma_I^n(x_{in}), \gamma_H^n(x_{in}), \mu_{VH}^n(x_{in})\}$, so that the cases are this way converted into rules. Once the *fuzzification* process concluded, the second step of the algorithm described in Section 3 is applied. This way, the rules that have variables with the same labels and belong to the same class are grouped together. In the example, it is observed that, among all the groups created, the one that has the greatest number of rules is shown in Table 1.

Table 1. Rules with common variables

| Rule | GNIm | GIPC | GIC | GTR | GTT | CON | SQUAL |
|------|------|------|-----|-----|-----|-----|-------|
| 1 | VL | VL | VH | VL | VL | VL | VL |
| 2 | VL | I | VH | VL | VL | VL | VL |
| 3 | VL | VH | VL | VL | VL | VL | VL |
| 4 | L | VL | VL | VL | VL | VL | VL |
| 5 | VL | VH | VH | VL | VL | VL | VL |

As can be seen, all the previous rules belong to the class SQUAL=VL and they have in common that GTR is {VL}, GTT is {VL} and CON is {VL}.

Analyzing rule 1, it can be seen that an amplification of the type ({VL, I, H, VH}, VL, VH, VL, VL, VL, VL) is possible, while an amplification of the type ({VL, L, I, H, VH}, VL, VH, VL, VL, VL, VL) is not possible since there is a rule (L, VL, VH, VL, VL, VL, L) in the initial repository that belongs to a different class. Continuing with this system of amplifications, the following rule is derived: ({VL, I, H, VH}, {VL, L, I, H, VH}, {VL, L, I, H, VH}, VL, VL, VL, VL), which can be expressed as follows:

If GNIm is not {L}, GTR is {VL}, GTT is {VL}, CON is {VL} then SQUAL is {VL}

This rule includes several rules of the initial set of rules. According to the algorithm, this process is repeated with the remaining initial rules that do not fulfil any of the amplified rules. In the example outlined, the following set of general rules is obtained:

- R₀: If GNIm is not {L}, GTR is {VL}, GTT is {VL}, CON is {VL} then SQUAL is {VL}.
- R₁: If GIPC is {H, VH} then SQUAL is {VL}.
- R₂: If GIC is {H, VH}, GTR is {L} then SQUAL is {VL}.
- R₃: If GNIm is not {H, VL}, GIPC is not {VH, H}, GIC is {VL}, GTR is {L, VL}, GTT is not {I, VH}, CON is {VL, L} then SQUAL is {VL}.
- R₄: If GNIm is not {VL, I}, GIC is {VH}, GTR is {VL} then SQUAL is {L}.
- R₅: If GTT is {I} then SQUAL is {I}.
- R₆: If GNIm is {L, H}, GIPC is {L, I}, GTT is not {I} then SQUAL is {H}.
- R₇: If GIPC is {VL, I}, CON is {H, VH} then SQUAL is {VH}.

At the beginning of the process of rule learning, 20% of the cases were reserved; they did not participate in the learning process with the aim of using them to check the validity of the obtained rules. When examining the rules that each case fulfils, it can be seen that there are no conflicts and that 100% of the cases are defined by the rules.

5 Conclusions

Collaborative learning support systems usually lack mechanisms to relate the form of collaborating of a group of students to the quality of the solution produced in this process. The use of fuzzy models allows calculating analysis indicators of collaboration with a vague semantics. In this article, a fuzzy model based on a machine learning process has been built to generate a set of rules as a result. The model has been put into practice using data registered in some collaborative problem-solving activities in the domain of Domotics. The rules obtained provide users not familiarized with this type of systems with information easy to understand about what type of collaboration leads to solutions of certain quality. Also, the easiness of computation of these rules allows its use on “real time” collaborative systems.

In the future, we aim at applying the algorithm proposed to databases that contain more extensive experiments, considering the possibility of using fuzzy temporal logic, being the temporary intervals those levels which the students go through during their learning. The implementation of facilities in CSCL environments to analyze in real time the users' interactions and to evaluate the collaborative activities according to the rules obtained can facilitate the generation of intelligent advice to the users in order to guide them to more fruitful situations of collaboration. The intelligent advice can be enriched and improved with the creation of fuzzy MIMO (Multiple Inputs, Multiple Outputs) models.

Acknowledgements. This work has been supported by the Consejería de Educación y Ciencia, Junta de Comunidades de Castilla-La Mancha (Spain), in the PBI-05-006 project.

References

1. Baker, M., Lund, K.: Promoting reflective interactions in a CSCL environment. *Journal of Computer Assisted Learning* 3 (13), (1997) 175-193.
2. Barros, B., Verdejo, M.F.: Analysing student interaction processes in order to improve collaboration. The DEGREE approach. *International Journal of Artificial Intelligence in Education*, 11, (2000) pp. 221-241.
3. Bravo, C., Redondo, M.A., Ortega, M. and Verdejo, M.F.: Collaborative environments for the learning of design: A model and a case study in Domotics. *Computers and Education* 46 (2), (2006) 152-173.
4. Castro, J.L., Castro-Schez, J.J., Zurita, J.M.: Learning maximal structure rules in fuzzy logic for knowledge acquisition in expert systems. *Fuzzy Sets and Systems*, (1999) 331-342.
5. Constantino-González, M.A., Suthers, D.D., Escamilla de los Santos, J.G.: Coaching web-based collaborative learning based on problem solution differences and participation. *International Journal of Artificial Intelligence in Education* 13, (2003) 263-299.
6. Dimitrakopoulou, A., et al.: State of the Art on Interaction Analysis: Interaction Analysis Indicators. Kaleidoscope Network of Excellence. *Interaction & Collaboration Analysis' Supporting Teachers and Students' Self-Regulation*. Jointly Executed Integrated Research Project. Deliverable D.26.1, (2004).

7. Gutwin, C., Stark, G., Greenberg, S.: Support for workspace awareness in educational groupware. Proceedings of CSCL'95. The First International Conference on Computer Support for Collaborative Learning, (1995) 147-156.
8. Holland, J.H., Reitman, J.S.: Cognitive systems based on adaptative algorithms. In Waterman, D. A. and Hayes-Roth, F., editors, Pattern-Directed Inference Systems. Academic Press, (1978).
9. Jenlink, P., Carr, A.A.: Conversation as a medium for change in education. Educational Technology, 3138 (1996).
10. Jermann, P., Soller, A., Muehlenbrock, M.: From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning. In the Proceedings of the European Conference on ComputerSupported Collaborative Learning, Maastricht, The Netherlands (2001).
11. Koshman, T.: CSCL: Theory and practice of an emerging paradigm. Hillsdale, NJ: Lawrence Erlbaum, (1996).
12. McManus, M., Aiken, R. "Monitoring computer-based problem solving," Journal of Artificial Intelligence in Education, 6(4), (1995) 307-336.
13. Muehlenbrock, M., Hoppe, U.: 'Computer supported interaction analysis of group problem solving'. Computer Support for Collaborative Learning (CSCL'1999). Palo alto, CA, USA, (1999) 398-405.
14. Redondo, M.A., Bravo, C., Bravo, J., Ortega, M.: Applying Fuzzy Logic to Analyze Collaborative Learning Experiences. Journal of the United States Distance Learning Association. Special issue on Evaluation Techniques and Learning Management Systems, (2003) Vol. 17, N° 2, pp 19-28.
15. Rosatelli, M., Self, J., and Thirty, M.: LeCs: A collaborative case study system. Proceedings of the 5th International Conference on Intelligent Tutoring Systems, Montreal, Canada, (2000) 242-251.
16. Slavin, R.E.: Cooperative learning: Theory, research and practice. Needham Heights, MA: Allyn and Bacon, (1995).
17. Soller A., Wiebe, J., Lesgold, A.: A Machine Learning Approach to Assessing Knowledge Sharing During Collaborative Learning Activities. Proceedings of Computer Support for Collaborative Learning (2002).
18. Urretavizcaya, M.: Sistemas inteligentes en el ámbito de la educación. Revistalberoamericana de Inteligencia Artificial. N° 12, (2001) pp. 5-12.
19. Wenger, E.: Artificial Intelligence and Tutoring Systems. Los Altos, CA: Morgan Kaufmann (1987).
20. Zadeh, L.A.: Fuzzy sets. Information and Control 8 (3), (1965) 338-353.

Electromagnetic Levitation System with Clustering Based Fuzzy Controller

Min-Soo Kim and Yeun-Sub Byun

Maglev Train System Research Team, Korea Railroad Research Institute
360-1 Woram-dong, Uiwang-si, Gyeonggi-do, Korea
ms_kim@krri.re.kr,
ysbyun@krri.re.kr

Abstract. This paper describes the development of a clustering based fuzzy controller of an electromagnetic suspension vehicle using gain scheduling method and Kalman filter for a simplified single magnet system. Electromagnetic suspension vehicle systems are highly nonlinear and essentially unstable systems. For achieving the levitation control, we considered a fuzzy system modeling method based on clustering algorithm which a set of input/output data is collected from the well defined Linear Quadratic Gaussian (LQG) controller. Simulation results show that the proposed clustering based fuzzy controller robustly yields uniform performance over the mass variation range.

1 Introduction

Maglev vehicles constitute a new class of transport systems that has been constantly developed and improved to become a new alternative of comfortable and secure transport. These vehicles have suspension, propulsion and guidance systems based on magnetic forces. A controlled DC electromagnetic suspension system is a highly nonlinear position regulator but an adequate insight into the design requirements can be obtained by considering a linear model. Different schemes of stabilization and control of single-degree of freedom suspension systems have been extensively studied at recent years [1][2].

In this paper, we suggest a design procedure of the clustering based fuzzy controller for electromagnetically levitated vehicle based on gain scheduling method and LQG regulator.

1. Selection of scheduling variables. In this paper, disturbance is chose as a scheduling variable
2. Linearization of the nonlinear plant model based on the scheduling variable.
3. Determination of a control law which composes the state variables and the scheduling variable.
4. Design of a LQG regulator. The LQG regulator consists of an optimal state-feedback gain and a Kalman state estimator[3].
5. Controller design using clustering based fuzzy logic based on the I/O data of LQG regulator.

This paper is organized as follows. Section 2 discusses concisely the single electromagnet levitation systems. Sect. 3 describes the structure of the levitation control system. Sect. 4 overviews the clustering based fuzzy system modeling method and Sect. 5 shows the simulation results. The main conclusions are then summarized in Sect. 6.

2 Modeling of the Single Magnet Levitation Systems

Each levitation electromagnet has an “U” shape as shown in Fig. 1 and its attraction force is given by

$$F(i, z) = \frac{\mu_0 N^2 A i^2(t)}{4z^2(t)} \tag{1}$$

where $i(t)$ is the current, $z(t)$ is the air gap and N is the number of turns.

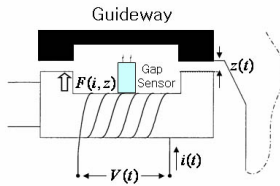


Fig. 1. Single magnet levitation system

If R is the total resistance of the circuit then for an instantaneous voltage $v(t)$ across the magnet winding, the excitation current is controlled by

$$v(t) = Ri(t) + \frac{\mu_0 N^2 A}{2z(t)} \frac{di(t)}{dt} - \frac{\mu_0 N^2 Ai(t)}{2z^2(t)} \frac{dz(t)}{dt} \tag{2}$$

With ignoring the flux leakage and reluctance of electromagnet, we can get the nonlinear dynamic model equation as Eq. (3) and (4).

$$m\ddot{z}(t) = -F(i, z) + f_d(t) + mg \tag{3}$$

$$\dot{i}(t) = \frac{\dot{z}(t)i(t)}{z(t)} + \frac{2z(t)}{\mu_0 N^2 A} (v(t) - Ri(t)) \tag{4}$$

where $f_d(t)$ is disturbance input, g is a gravity constant, A is cross-section area, and m is the mass of the suspended object.

By choosing $z(t)$, $\dot{z}(t)$ and $i(t)$ as the state variables, the nonlinear state-space representation of the above equation is

$$\begin{aligned} \dot{x}(t) &= f[x(t), u(t), f_d(t)] \\ y(t) &= h[x(t)] \end{aligned} \tag{5}$$

where $\dot{x}_1(t) = x_2(t)$

$$\dot{x}_2(t) = -\frac{1}{m} \frac{\mu_0 N^2 A}{4} \frac{x_3(t)^2}{x_1(t)^2} + g + \frac{1}{m} f_d(t)$$

$$y(t) = x_1(t)$$

$$x^T(t) = [x_1(t), x_2(t), x_3(t)] = [z(t), \dot{z}(t), i(t)]$$

$$\dot{x}_3(t) = \frac{x_2(t)x_3(t)}{x_1(t)} + \frac{2x_1(t)}{\mu_0 N^2 A} (u(t) - Rx_3(t))$$

$$u(t) = v(t)$$

3 Structure of the Proposed Levitation Control System

The feedback system of the proposed levitation control is depicted schematically in Fig. 2.

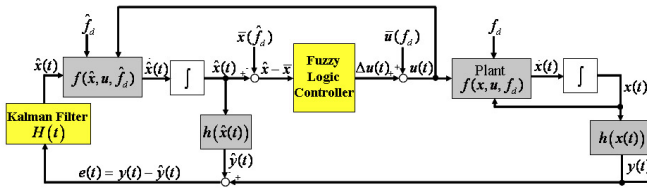


Fig. 2. Block diagram of the proposed levitation control system

At the first step of the design process, disturbance $f_d(t)$ is selected for scheduling variable which has limited boundary as $F_d := \{f_d(t) \mid f_d(t) \in [f_d^-, f_d^+], t \geq 0\}$. And the state vectors of a operating point $\bar{x}(\bar{f}_d)$ is expressed as

$$\bar{x}(\bar{f}_d) = [z_0 \quad 0 \quad i_0] \quad , \quad i_0 = z_0 \sqrt{\frac{4(mg + \bar{f}_d)}{\mu_0 N^2 A}} \quad , \quad \bar{f}_d \in F_d \tag{6}$$

From the optimal feedback gain matrix K , the control law which composes the state variables and the scheduling variable is given by

$$u(t) = K(x(t), f_d(t)) \tag{7}$$

where $K(\bullet)$ is a smoothing function which makes the state vectors of operating point $\bar{x}(\bar{f}_d)$ constant values in the closed-loop system and can be a stable systems.

The existence of the operating point $\bar{x}(\bar{f}_d)$ in the closed-loop systems means that the smoothing function should satisfy the followings:

$$\bar{u}(\bar{f}_d) = K(\bar{x}(\bar{f}_d), \bar{f}_d) = Rz_0 \sqrt{\frac{4(mg + \bar{f}_d)}{c}} \quad , \quad c = \mu_0 N^2 A \tag{8}$$

$$y(\bar{f}_d) = z_0 \tag{9}$$

Then the linearization model, as the second step of the design process, through the scheduling variable from the nonlinear state-space equations is approximately followings:

$$\begin{aligned}
 \Delta \dot{x}(t) &= A(\bar{f}_d)\Delta x(t) + B\Delta u(t) + D\Delta f_d(t) \\
 \Delta u(t) &= K_1^T(\bar{f}_d)\Delta x(t) + K_2^T(\bar{f}_d)\Delta f_d(t) \\
 \Delta \hat{y}(t) &= C\Delta x(t)
 \end{aligned}
 \tag{10}$$

where $\Delta x(t) = x(t) - \bar{x}(\bar{f}_d)$, $\Delta u(t) = u(t) - \bar{u}(\bar{f}_d)$, $\Delta f_d(t) = f_d(t) - \bar{f}_d$

$$\begin{aligned}
 A(\bar{f}_d) = \frac{\partial f}{\partial x} \bar{x}(\bar{f}_d), \bar{u}(\bar{f}_d), \bar{f}_d &= \begin{bmatrix} 0 & 1 & 0 \\ \frac{2(mg + \bar{f}_d)}{mz_0} & 0 & -\sqrt{\frac{c(mg + \bar{f}_d)}{mz_0}} \\ 0 & \sqrt{\frac{4(mg + \bar{f}_d)}{c}} & -\frac{2Rz_0}{c} \end{bmatrix} \\
 B^T = \frac{\partial f}{\partial x} \bar{x}(\bar{f}_d), \bar{u}(\bar{f}_d), \bar{f}_d &, \quad C = [1 \ 0 \ 0] \ , \quad D^T = \frac{\partial f}{\partial x} \bar{x}(\bar{f}_d), \bar{u}(\bar{f}_d), \bar{f}_d \\
 = \begin{bmatrix} 0 & 0 & \frac{2z_0}{c} \end{bmatrix} &, \quad = \begin{bmatrix} 0 & \frac{1}{m} & 0 \end{bmatrix}
 \end{aligned}$$

From Rugh's method [4] and Eq. (10), the control laws $u(t)$ as the third step of the design process is computed by

$$u(t) = K_1^T(\hat{f}_d(t)) [x(t) - \bar{x}(\hat{f}_d(t))] + \bar{u}(\hat{f}_d(t))
 \tag{11}$$

The disturbance signal containing the scheduling variable can be expressed as

$$\hat{f}_d(t) = m_0 \ddot{z}(t) + \frac{\mu_0 N^2 A}{4} \left(\frac{i(t)}{z(t)} \right)^2 - m_0 g
 \tag{12}$$

The levitation control laws based on a fuzzy logic control (FLC) as a smoothing function $K(\bullet)$ and Kalman filter $H(t)$ as a state estimator is used to improve system reliability. That is, the controller inputs $u(t)$ is evaluated using the difference between estimated state variables $\hat{x}(t)$ from Kalman filter and state variables at operating point $\bar{x}(t)$ in Eq. (6). As a smoothing function, a fuzzy logic control (FLC) based on the data clustering algorithm extracted the I/O data from LQG regulator is used.

In following section, the clustering based fuzzy system modeling method will be described.

4 Clustering Based Fuzzy System Modeling

4.1 Modeling of Fuzzy System

Modeling fuzzy systems involves identification of the structure and the parameters with given training data. In the Sugeno fuzzy model[5], unlike the Mamdani method[6], the consequent part is represented by a linear or nonlinear function of input variables. Fuzzy rules in the MISO Sugeno model with n inputs x_1, \dots, x_n and an output variable y_i of the i th fuzzy rule is of the form:

$$\text{IF } x_1 \text{ is } A_{1i} \text{ and } \dots \text{ and } x_n \text{ is } A_{ni}, \text{ THEN } y_i = f_i(x_1, \dots, x_n)
 \tag{13}$$

where $i=1, \dots, M$ and A_{ij} is a linguistic label represented by the membership function $A_{ij}(x_j)$, and $f_i(\cdot)$ denotes a function that relates input to output.

$$A_{ij}(x_j) = \exp\left(-\frac{1}{2}\left(\frac{x_j - c_{ij}}{w_{ij}}\right)^2\right) \tag{14}$$

$$f_i(x_1, \dots, x_n) = a_{0i} + a_{1i}x_1 + \dots + a_{ni}x_n \tag{15}$$

where the parameters c_{ij} and w_{ij} define center and width of the Gaussian membership function $A_{ij}(x_j)$. The coefficients $a_{0i}, a_{1i}, \dots, a_{ni}$ are to be determined from input-output training data. In the simplified reasoning method, the output y of the fuzzy system with M rules is represented as

$$y = \frac{\sum_{i=1}^M \mu_i f_i(x_1, \dots, x_n)}{\sum_{i=1}^M \mu_i} \tag{16}$$

where μ_i is a degree of relevance. In the product implication method, the degree of relevance is defined as

$$\mu_i = \prod_{j=1}^n A_{ij}(x_j) = \exp\left(-\frac{1}{2} \sum_{j=1}^n \left(\frac{x_j - c_{ij}}{w_{ij}}\right)^2\right) \tag{17}$$

In the Sugeno fuzzy model, the time-consuming rule extraction process from experience of human experts or engineering common sense reduces to a simple parameter optimization process of coefficients $a_{0i}, a_{1i}, \dots, a_{ni}$ for a given input-output data set since the consequent part is represented by a linear function of input variables. Using linear function in the consequent part, a group of simple fuzzy rules can successfully approximate nonlinear characteristics of practical complex systems.

4.2 Clustering Algorithm

The parameters of the component fuzzy systems are characterized using clustering algorithms. The subtractive clustering algorithm[7] finds cluster centers $x_i^* = (x_{1i}^*, \dots, x_{ni}^*)$ of data in input-output product space by computing the potential values at each data point.

There are n -dimensional input vectors x_1, x_2, \dots, x_m and 1-dimensional outputs y_1, y_2, \dots, y_m forming $(n+1)$ -dimensional space of input-output data. For data X_1, X_2, \dots, X_N in $(n+1)$ -dimensional input-output space, the subtractive clustering algorithm produces cluster centers as in the following procedure:

Step 1: Normalize given data into the interval [0,1].

Step 2: Compute the potential values at each data point. The potential value P_i of the data X_i is computed as

$$P_i = \sum_{j=1}^N \exp(-\alpha \|X_i - X_j\|^2), \quad i = 1, 2, \dots, N \tag{18}$$

where a positive constant $\alpha = 4/r_a^2$ determines the data interval which affects the potential values. Data outside the circle with radius over positive constant $r_a < 1$ do not substantially affect potential values.

Step 3: Determine the data with the largest potential value P_1^* as the first cluster center X_1^* .

Step 4: Compute the potential value P_i' after eliminating the influence of the first cluster center.

$$P_i' = P_i - P_1^* \sum_{j=1}^N \exp(-\beta \|X_i - X_1^*\|^2) \tag{19}$$

where positive constant $\beta = 4/r_b^2$ prevents the second cluster center from locating close to the first cluster center. If the effect of potential of the first cluster center is not eliminated, second cluster center tends to appear close to the first cluster center, since there are many data concentrated in the first cluster center. Taking $r_b > r_a$ makes the next cluster center not appear near the present cluster center.

Step 5: Determine the data point of the largest potential value P_2^* as the second cluster center X_2^* . In general, compute potential values P_i' after removing the effect of the k th cluster center X_k^* , and choose the data of the largest potential value as the cluster center X_{k+1}^*

$$P_i' = P_i - P_k^* \exp(-\beta \|X_i - X_k^*\|^2) \tag{20}$$

Step 6: Check if we accept the computed cluster center. If $P_k^*/P_1^* \geq \bar{\epsilon}$, or $P_k^*/P_1^* > \underline{\epsilon}$ and $d_{\min}/r_a + P_k^*/P_1^* \geq 1$, then accept the cluster center and repeat step 5. Here d_{\min} denotes the shortest distance to the cluster centers $X_1^*, X_2^*, \dots, X_k^*$ determined so far. If $P_k^*/P_1^* > \underline{\epsilon}$ and $d_{\min}/r_a + P_k^*/P_1^* < 1$, then set the X_k^* to 0 and select the data of the next largest potential. If $d_{\min}/r_a + P_k^*/P_1^* \geq 1$ for the data, choose this data as the new cluster center and repeat step 5. If $P_k^*/P_1^* \leq \underline{\epsilon}$, terminate the iteration.

Fuzzy system modeling process using the cluster centers $X_1^*, X_2^*, \dots, X_M^*$ in input-output space is as follows. The input part of the cluster centers corresponds to antecedent fuzzy sets. In $(n+1)$ -dimensional cluster center X_i^* , the first n values are n -dimensional input space $x_i^* = (x_{i1}^*, \dots, x_{in}^*)$. Each component determines the center of membership functions for each antecedent fuzzy sets. The cluster centers become the

center of the membership functions $c_{ij} = x_{ij}^*$. The width of the membership function w_{ji} is decided as $w_{ij} = r_a \left\| \max_i(x_i^*) - \min_i(x_i^*) \right\| / \sqrt{M}$ where M denotes the number of cluster centers, $\left\| \max_i(x_i^*) - \min_i(x_i^*) \right\|$ denotes the difference between the maximum and the minimum distances between cluster centers. The number of cluster centers corresponds to the number of fuzzy rules. The next process is to compute optimal consequent parameters $a_{0i}, a_{1i}, \dots, a_{ni}$ in order to produce output y_j of the y_j th rule in the Sugeno fuzzy model. The number of centers equals the number of fuzzy rules. The output of the fuzzy system is defined as a linear function of input variables.

$$y_i = a_{0i} + a_{1i}x_1 + a_{2i}x_2 + \dots + a_{ni}x_n = a_i^T x + a_{0i} \tag{21}$$

Compute parameters g_i through linear least-squares estimation, the final output y of the Sugeno fuzzy model is given as

$$y = \frac{\sum_{i=1}^M \mu_i (a_i^T x + a_{0i})}{\sum_{i=1}^M \mu_i} \tag{22}$$

This is the final output of the fuzzy system.

5 Simulation and Results

Electromagnet used for this paper was previously used on a research vehicle which consists of 3 bogies and 24 electromagnets insulated copper winding wound on steel cores and design to lift a maximum load of 10 times of electromagnet mass at a nominal operating air gap of 10 [mm]. An empty vehicle weights 22 [ton] and full loaded vehicle weights 28 [ton].

The closed-loop responses for a reference gap of 10 [mm] at initial gap of 17 [mm] has been considered. A step disturbance (30% of the nominal vehicle weight) between 0.5 [sec] and 1 [sec], and measurement noise (5% of the nominal gap as a random noise) have been applied to the plant by suddenly laying a mass on the vehicle. Nominal vehicle weight 1,041.7 [kg](= 26[ton]/24) is used.

The disturbance step input is presented in Fig. 3. And Fig. 4 shows that the transient gap responses exhibits containing a disturbance and measurement noise. The disturbance has been completely rejected in steady-state. Fig. 5 shows the control input as a function of time.

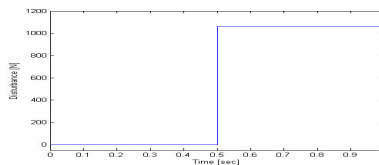


Fig. 3. Disturbance step input

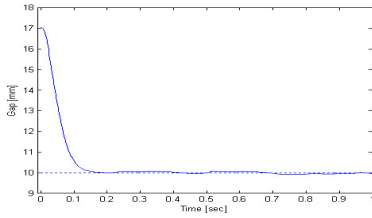


Fig. 4. Closed-loop response of nonlinear plant at the nominal point

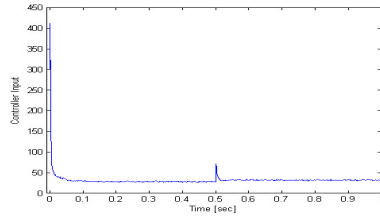


Fig. 5. Control input of nonlinear plant at the nominal point

Fig. 6 and Fig. 7 show the time response and the control inputs compared with the mass variation range which is nominal mass, full load, and empty load of vehicle, respectively.

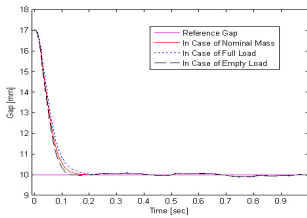


Fig. 6. Comparison of time responses over the mass variation range

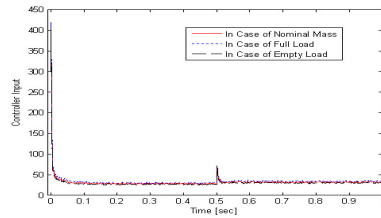


Fig. 7. Comparison of control inputs over the mass variation range

Table 1 shows the simulation results of the Root Mean Square Error (RMS) to the gap response and control input for conventional LQG and FLC, respectively.

Table 1. Experimental results of Root Mean Square Error (RMS)

| | LQG | | | Fuzzy | | |
|-----------------------|---------|--------|-----------|---------|--------|-----------|
| | nominal | empty | full load | nominal | empty | full load |
| RMS for response | 1.3613 | 1.3168 | 1.4056 | 1.3643 | 1.4167 | 1.3252 |
| RMS for control input | 35.45 | 37.81 | 33.29 | 35.71 | 37.89 | 33.71 |

In above Fig. 6, Fig. 7 and Table 1, we can confirm that the time responses and control inputs are nearly invariant under all mass variations. Therefore we can conclude that the proposed fuzzy system and Kalman filter based on LQG I/O data achieves the robust stability and performance under mass variations.

6 Conclusions

Gain scheduling is one of the most popular design methods for a various control system. Fuzzy control approach to the levitation of an electromagnetic suspension

vehicle prototype based on gain scheduling coupling LQG and Kalman filter has been proposed in this paper. The proposed design method is very useful because it easily obtains from various I/O data designed by classical control method.

Simulation results have been shown that the proposed gain scheduling method based on FLC and continuous Kalman filter methodology robustly shows reasonable performance with adequate gap response over the mass variation range.

References

1. Sinha, P. K.: Electromagnetic suspension: dynamics and control, Peter Peregrinus Ltd, London (1997)
2. Jayawant, B. V.: Electromagnetic suspension and levitation, Proceedings of the IEE, Vol. 129, Pt. A, No. 8 (1992), pp. 549-581
3. Athans, M.: A tutorial on the LQG/LTR Method, Proc. ACC (1986)
4. Rugh, W. J.: Analytical framework for gain scheduling, IEEE control systems Vol. 11, No. 1 (1991) pp. 79-84
5. Sugeno, M.: Industrial Applications of Fuzzy Control, Elsevier Science Pub., (1985).
6. Mamdani, E. H. and Assilian, S. :An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller, Int. J. of Man Machine Studies, Vol. 7, No. 1,(1975), pp.1-13.
7. Chiu S.: Fuzzy Model Identification Based on Cluster Estimation, Journal of Intelligent & Fuzzy Systems, Vol. 2, No. 3 (1994)
8. Kim, M. S., Byun, Y. S., Lee, Y. H., and Lee, K. S.: Gain Scheduling Control of Levitation System in Electromagnetic Suspension Vehicle, WSEAS Trans. on Circuits and Systems, Vol. 5, (2006), pp. 1706-1712.

Fuzzy Relation-Based PNNs with the Aid of IG and Symbolic Gene Type-Based GAs

Sung-Kwun Oh¹, In-Tae Lee¹, Hyun-Ki Kim¹, and Seong-Whan Jang²

¹ Department of Electrical Engineering, The University of Suwon, San 2-2, Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea
{ohsk, hkkim}@suwon.ac.kr

² Department of Electrical Electronic and Information Engineering, Wonkwang University, 344-2, Shinyong-Dong, Iksan, Chon-Buk, 570-749, South Korea
swhjang@wonkwang.ac.kr

Abstract. In this paper, we propose a new design methodology of fuzzy-neural networks – Fuzzy Relation-based Polynomial Neural Networks (FRPNN) with symbolic genetic algorithms and Information Granules (IG). We have developed a design methodology based on symbolic genetic algorithms to find the optimal structure for fuzzy-neural networks that expanded from Group Method of Data Handling (GMDH). Such parameters as the number of input variables, the order of the polynomial, the number of membership functions, and a collection of the specific subset of input variables are optimized for topology of FRPNN with the aid of symbolic genetic optimization that has search capability to find the optimal solution on the solution space. The augmented and genetically developed FRPNN (gFRPNN) results in a structurally optimized structure and comes with a higher level of flexibility in comparison to the one we encounter in the conventional FRPNNs. The GA-based design procedure being applied at each layer of FRPNN leads to the selection of the most suitable nodes (or FRPNs) available within the FRPNN. The performance of genetically optimized FRPNN (gFRPNN) is quantified through experimentation where we use a number of modeling benchmarks data which are already experimented with in fuzzy or neurofuzzy modeling.

1 Introduction

It is expected that efficient modeling techniques should allow for a selection of pertinent variables and a formation of highly representative datasets. Furthermore, the resulting models should be able to take advantage of the existing domain knowledge (such as a prior experience of human observers or operators) and augment it by available numeric data to form a coherent data-knowledge modeling entity. Most recently, the omnipresent trends in system modeling are concerned with a broad range of techniques of computational intelligence (CI) that dwell on the paradigm of fuzzy modeling, neurocomputing, and genetic optimization [1, 2]. The list of evident landmarks in the area of fuzzy and neurofuzzy modeling [3, 4] is impressive. While the accomplishments are profound, there are still a number of open issues regarding structure problems of the models along with their comprehensive development and testing.

As one of the representative and advanced design approaches comes a family of fuzzy polynomial neuron (FPN or FRPN)-based self organizing neural networks (abbreviated as FRPNN or PNN and treated as a new category of neuro-fuzzy networks)[5-7]. The design procedure of FRPNNs exhibits some tendency to produce overly complex networks as well as comes with a repetitive computation load caused by the trial and error method being a part of the development process. The latter is in essence inherited from the original Group Method of Data handling (GMDH) algorithm that requires some repetitive parameter adjustment by the designer.

In this paper, to address the above design problems associated with the development of conventional FRPNN, we introduce the information granulation based genetically optimized FRPNN (IG_gFRPNN). We confine ourselves to the HCM clustering algorithm [10] to develop information granules, and subsequently construct the structure of the premise part and the consequent part of the fuzzy rules. To assess the performance of proposed model, we experiment with well-know medical imaging system (MIS) [11] widely used in software engineering.

2 The Architecture and Development of FRPNN

In this section, we elaborate on the architecture and a development process of the FRPNN. This network emerges from the genetically optimized multi-layer perceptron architecture based on GA algorithms and the extended GMDH method.

2.1 Fuzzy Relation Polynomial Neurons (FRPNs)

The FRPN encapsulates a family of nonlinear “if-then” rules. When put together, FRPNs results in a Fuzzy Relation—based Polynomial Neural Network (FRPNN). The FRPN consists of two basic functional modules. The first one, labeled by **F**, is a collection of fuzzy sets (here denoted by $\{A_l\}$ and $\{B_k\}$) that form an interface between the input numeric variables and the processing part realized by the neuron. Here x_q and x_p stand for input variables. The second module (denoted here by **P**) refers to the function – based nonlinear (polynomial) processing that involves some input variables.

In other words, FRPN realizes a family of multiple-input single-output rules.

$$\text{if } x_p \text{ is } A_l \text{ and } x_q \text{ is } B_k \text{ then } z \text{ is } P_{lk}(x_i, x_j, \mathbf{a}_{lk}) \tag{1}$$

Where \mathbf{a}_{lk} is a vector of the parameters of the conclusion part of the rule while $P_{lk}(x_i, x_j, \mathbf{a}_{lk})$ denotes the regression polynomial forming the consequence part of the fuzzy rule which uses several types of high-order polynomials. The activation levels of the rules contribute to the output of the FRPN being computed as a weighted average of the individual condition parts (functional transformations) P_K (note that the index of the rule, namely “ K ” is a shorthand notation for the two indexes of fuzzy sets used in the rule (1), that is $K = (l, k)$).

$$z = \sum_{K=1}^{all \text{ rules}} \mu_K P_K(x_i, x_j, \mathbf{a}_K) / \sum_{K=1}^{all \text{ rules}} \mu_K = \sum_{K=1}^{all \text{ rules}} \tilde{\mu}_K P_K(x_i, x_j, \mathbf{a}_K) \tag{2}$$

In the above expression, we use an abbreviated notation to describe an activation level of the “K”th rule to be in the form

$$\tilde{\mu}_K = \frac{\mu_K}{\sum_{L=1}^{all\ rules} \mu_L} \tag{3}$$

The FRPNN comes as a highly versatile architecture both with respect to the flexibility of the individual nodes (that are essentially banks of nonlinear “if-then” rules) as well as the interconnectivity occurring between the nodes and organization of the layers.

2.2 Genetic Algorithms-Based FRPNN

Let us briefly recall that GAs is a stochastic search technique based on the principles of evolution, natural selection, and genetic recombination by simulating a process of “survival of the fittest” in a population of potential solutions (individuals) to the given problem. GAs are aimed at the global exploration of a solution space. We are able to reduce the solution space with the aid of symbolic gene type genetic algorithms. That is the important advantage of symbolic gene type genetic algorithms. In order to enhance the learning of the FRPNN, we use GAs to complete the structural optimization of the network by optimally selecting such parameters as the number of input variables (nodes), the order of polynomial, and input variables within a FRPN.

In this study, GA uses the serial method of symbolic type, roulette-wheel used in the selection process, one-point crossover in the crossover operation, and a uniform mutation operator. To retain the best individual and carry it over to the next generation, we use elitist strategy.

3 The Design Procedure of Genetically Optimized FRPNN

We use FRPNNs as the building blocks of the network. Each neuron of the network realizes a polynomial type of partial description (PD) of the mapping between input and output variables. The input-output relation formed by the FRPNN algorithm can be described in the form

$$y = f(x_1, x_2, \dots, x_n) \tag{4}$$

The estimated output \hat{y} reads as the following polynomial

$$\hat{y} = \hat{f}(x_1, x_2, \dots, x_n) = c_0 + \sum_{k1} c_{k1}x_{k1} + \sum_{k1k2} c_{k1k2}x_{k1}x_{k2} + \sum_{k1k2k3} c_{k1k2k3}x_{k1}x_{k2}x_{k3} + \dots \tag{5}$$

where c_k s are its coefficients.

The framework of the design procedure of the Fuzzy Relation-based Polynomial Neural Networks (FRPNN) based on genetically optimized multi-layer perceptron architecture comprises the following steps

- [Step 1] Determine system’s input variables.
- [Step 2] Form training and testing data.
- [Step 3] Specify initial design parameters.
- [Step 4] Decide FRPN structure using genetic design.
- [Step 5] Carry out fuzzy inference and coefficient parameters estimation for fuzzy identification in the selected node (FRPN).

The regression fuzzy inference (reasoning scheme) is envisioned: The consequence part can be expressed by linear, quadratic, or modified quadratic polynomial equation. The use of the regression polynomial inference method gives rise to the expression

$$R^n : \text{If } x_1 \text{ is } A_{n1} \text{ and } \dots \text{ and } x_k \text{ is } A_{nk} \text{ then} \tag{6}$$

$$y_n - M_n = f_n \{ (x_1 - v_{n1}), (x_2 - v_{n2}), \dots, (x_k - v_{nk}) \}$$

Where, R^n is the n -th fuzzy rule, $x_l(l=1, 2, \dots, k)$ is an input variable, $A_{jl}(j=1, \dots, n; l=1, \dots, k)$ is a membership function of fuzzy sets, $v_{jl}(j=1, \dots, n; l=1, \dots, k)$ is the center point related to the new created input variable, $M_j(j=1, \dots, n)$ is the center point related to the new created output variable, n denotes the number of the rules, $f_l(\cdot)$ is a regression polynomial function of the input variables as shown in Table 1.

The coefficients of consequence part of fuzzy rules obtained by least square method (LSE)

- [Step 6] Select nodes (FRPNs) with the best predictive capability and construct their corresponding layer.
- [Step 7] Check the termination criterion.

The termination condition that controls the growth of the model consists of two components, that is the performance index and a size of the network (expressed in terms of the maximal number of the layers). As far as the performance index is concerned (that reflects a numeric accuracy of the layers), a termination is straightforward and comes in the form

$$F_1 \leq F_* \tag{7}$$

Where, F_1 denotes a maximal fitness value occurring at the current layer whereas F_* stands for a maximal fitness value that occurred at the previous layer.

In this study, we use performance index that is the Mean Squared Error (MSE).

$$E(PI \text{ or } EPI) = \frac{1}{N} \sum_{p=1}^N (y_p - \hat{y}_p)^2 \tag{8}$$

where, y_p is the p -th target output data and \hat{y}_p stands for the p -th actual output of the model for this specific data point. N is training(PI) or testing(EPI) input-output data pairs and E is an overall(global) performance index defined as a sum of the errors for the N .

- [Step 8] Determine new input variables for the next layer.

The outputs of the preserved nodes ($z_{1i}, z_{2i}, \dots, z_{wi}$) serve as new inputs to the next layer ($x_{1j}, x_{2j}, \dots, x_{wj}$)($j=i+1$). This is captured by the expression

$$x_{1j} = z_{1i}, x_{2j} = z_{2i}, \dots, x_{wj} = z_{wi} \quad (9)$$

The FRPNN algorithm is carried out by repeating steps 4-8.

4 Experimental Studies

In this section, we illustrate the development of the IG_gFRPNN and show its performance for well known and widely used datasets in software engineering, medical imaging system (MIS) dataset[11].

We consider a medical imaging system (MIS) subset of 390 software modules written in Pascal and FORTRAN for modeling. These modules consist of approximately 40,000 lines of code. To design an optimal model from the MIS, we study 11 system input variables such as LOC, CL, TChar, TComm, MChar, DChar, N, NE, N_F, V(G) and BW. The output variable of the model is the number of changes *changes* made to the software module during its development. In the case of the MIS data, the performance index is defined as the mean squared error (MSE) as in (8).

Table 1 summarizes the list of parameters used in the genetic optimization of the network. In the optimization of each layer, we use 150 generations, 300 populations, crossover rate equal to 0.65, and the probability of mutation set up to 0.1. A chromosome used in the genetic optimization consists of a string including 4 sub-chromosomes. The first chromosome contains the number of input variables, the second chromosome contains input variables, the third chromosome contains number of membership functions, and finally the fourth chromosome the order of polynomial.

Table 2 summarizes the performance of the 1st to 3rd layer of the network when changing the maximal number of inputs; here the "Max" parameter was set up to 2 through 3. Here, node "0" indicates that it has not been selected by the genetic operation. In "Node (MFs)" of Table 2, Node stands for node number while MFs means number of membership functions per each input variable.

In the nodes (FRPNs) of Fig. 1, 'FRPN_n' denotes the nth FRPN (node) of the corresponding layer, the number of the left side denotes the number of nodes (inputs or FRPNs) coming to the corresponding node, and the number of the right side denotes the polynomial order of conclusion part of fuzzy rules used in the corresponding node. And the number located in the rectangle in front of each node means no. of MFs.

Table 3 summarizes a comparative analysis of the performance of the network with other models. The experimental results clearly reveal that it outperforms the existing models both in terms of significant approximation capabilities (lower values of the performance index on the training data, PI_s) as well as superb generalization abilities (expressed by the performance index on the testing data EPI_s). To effectively reduce a large number of nodes and avoid a large amount of time-consuming iteration of FRPNN, the stopping criterion can be taken into consideration up to maximally the 1st or 3rd layer. Therefore the width (the number of nodes) of its layer as well as the depth (the number of layers) of the proposed Information Granulation and genetically optimized FRPNN (IG_gFRPNN) can be much lower in comparison to the

Table 1. Summary of the parameters of the genetic optimization

| Parameters | | 1 st to 3 rd layer |
|----------------------|--|--|
| GA | Maximum generation | 150 |
| | Total population size | 300 |
| | Selected population size (<i>W</i>) | 30 |
| | Crossover rate | 0.65 |
| | Mutation rate | 0.1 |
| | String length | 1+(Max*2)+1 |
| FRPNN | Maximal no.(Max) of inputs to be selected | 1≤T≤Max(2~3) |
| | Polynomial type (Type T) of the consequent part of fuzzy rules | 1≤T≤4 |
| | Consequent input type to be used for Type T (*) | Type T*,T |
| | Membership Function (MF) type | Triangular Gaussian-like |
| No. of MFs per input | | 2 or 3 |

l, T, Max: integers, Type T* means that entire system inputs are used for the polynomial in the conclusion part of the rules.

Table 2. Performance index of the network of each layer versus the increase of maximal number of inputs to be selected

(a) Triangular MF

| M A X | 1 st layer | | | 2 nd layer | | | 3 rd layer | | |
|-------------|-----------------------|---|---------------|-----------------------|---|---------------|-----------------------|---|---------------|
| | Node (MFs) | T | PI EPI | Node (MFs) | T | PI EPI | Node (MFs) | T | PI EPI |
| 2 | 4(3) 10(2) | 1 | 58.277 29.968 | 2(2) 3(3) | 4 | 19.368 22.346 | 4(2) 23(2) | 3 | 17.302 18.228 |
| 3 | 4(3) 10(2) 0 | 1 | 58.277 29.968 | 2(2) 3(3) 0 | 4 | 19.368 22.346 | 2(2) 23(2) 0 | 4 | 18.049 18.986 |

(b) Gaussian-like MF

| M A X | 1 st layer | | | 2 nd layer | | | 3 rd layer | | |
|-------------|-----------------------|---|---------------|-----------------------|---|---------------|-----------------------|---|---------------|
| | Node (MFs) | T | PI EPI | Node (MFs) | T | PI EPI | Node (MFs) | T | PI EPI |
| 2 | 8(2) 10(2) | 1 | 67.601 39.644 | 17(2) 21(2) | 3 | 36.682 32.250 | 2(2) 29(2) | 4 | 33.815 29.636 |
| 3 | 5(3) 7(3) 10(2) | 2 | 49.612 35.485 | 1(2) 11(2) 0 | 2 | 45.291 29.693 | 2(2) 11(2) 0 | 1 | 48.110 23.172 |

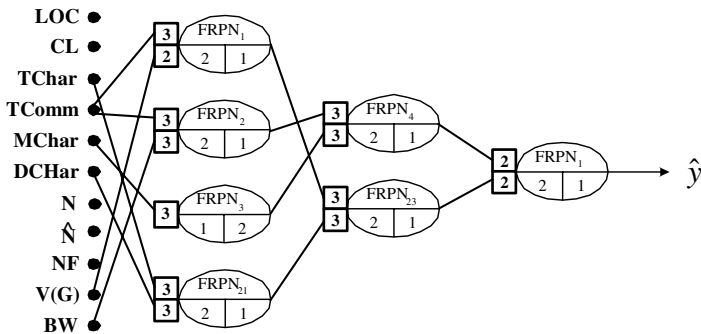


Fig. 1. Optimal networks structure of IG_gFRPNN (for 3 layers)

Table 3. Comparative analysis of the performance of the network; considered are models reported in the literature

| Model | Structure | | | PI | EPI | |
|---------------|------------------|--------------|--------------------|-----------------------------|--------|--------|
| SONFN [12] | Simplified | Generic Type | Basic Architecture | 5 th layer Case1 | 40.753 | 17.898 |
| | Linear | Generic Type | Basic Architecture | 5 th layer Case2 | 35.745 | 17.807 |
| FPNN [13] | No. of inputs: 2 | Triangular | Type : 2 | 5 th layer | 32.195 | 18.462 |
| | | Gaussian | Type : 1 | 5 th layer | 49.716 | 31.423 |
| | No. of inputs: 3 | Triangular | Type : 1 | 5 th layer | 32.251 | 19.622 |
| | | Gaussian | Type : 1 | 5 th layer | 39.093 | 19.983 |
| Our model | max-inputs: 2 | Triangular | | 3 rd layer | 17.302 | 18.228 |
| | | Gaussian | | 3 rd layer | 33.815 | 29.636 |
| | max-inputs: 3 | Triangular | | 3 rd layer | 18.049 | 18.986 |
| | | Gaussian | | 3 rd layer | 48.110 | 23.172 |

conventional optimized FPNN as shown in Table 3 (which immensely contributes to the compactness of the resulting network). $PI_s(EPI_s)$ is defined as the mean square errors (MSE) computed for the experimental data and the respective outputs of the network.

5 Concluding Remarks

In this paper, we have introduced and investigated a class of information granulation based genetically optimized Fuzzy Relation Polynomial Neural Networks (IG_gFRPNN).

The GA-based design procedure applied at each stage (layer) of the FRPNN leads to the selection of the preferred nodes (or FRPNs) with some well-defined optimal local characteristics (such as the number of input variables, the order of the consequent polynomial of fuzzy rules, a collection of the subset of input variables, and the number of membership functions). These options contribute to the flexibility of the resulting architecture of the network. The design methodology supports a hybrid structural optimization (based on GMDH method and genetic optimization) followed by some parametric learning. The GMDH method is augmented by the structural phase supported by some evolutionary optimization that is followed by the phase of the Least Square Estimation (LSE).

The comprehensive experimental studies involving well-known datasets demonstrate a superb performance of the network when compared to the existing fuzzy and neurofuzzy models. More importantly, through the proposed framework of genetic optimization we can efficiently search for the optimal network architecture (being both structurally and parametrically optimized) and this design facet becomes crucial in improving the overall performance of the resulting model.

As the polynomial neural networks have been developed in different ways, endowed with numerous learning mechanisms of structural and parametric learning, it is helpful to establish a global perspective as to the evolution of the area and identify

where the current research pursuits are firmly positioned. Most importantly, the proposed framework of genetic optimization supports an efficient structural search resulting in the structurally and parametrically optimal architectures of the networks.

Acknowledgements. This work was supported by Wonkwang University (2006).

References

1. Pedrycz, W.: Computational Intelligence: An Introduction, CRC Press, Florida, (1998)
2. Peters, J.F, Pedrycz, W.: Computational intelligence, In Encyclopedia of Electrical and Electronic Engineering, (Edited by J.G. Webster). John Wiley & Sons, New York. 22 (1999)
3. Pedrycz, W., Peters, J.F. (ed.): Computational Intelligence in Software Engineering, World Scientific, Singapore. (1998)
4. Takagi, H., Hayashi, I.: NN-driven fuzzy reasoning, Int. J. of Approximate Reasoning. 5 (3) (1991), 191-212
5. Cordon, O., et al.: Ten years of genetic fuzzy systems: current framework and new trends, Fuzzy Sets and Systems. 141 (1) (2004) 5-31
6. Oh, S.K., Pedrycz, W.: "Self-organizing Polynomial Neural Networks Based on PNs or FPNs : Analysis and Design", Fuzzy Sets and Systems. 142 (2004) 163-198
7. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, Berlin Heidelberg. (1996)
8. D. E. Goldberg, Genetic Algorithm in search, Optimization & Machine Learning, Addison wesley, 1989.
9. K. De Jong. Are genetic algorithms function optimizers? In Proc. of PPSN II (Parallel Problem Solving from Nature), pages 3-13, Amsterdam, North Holland, 1992.
10. J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, New York, Plenum, 1981.
11. Lyu, M.R. (ed.): Handbook of Software Reliability Engineering , McGraw-Hill. (1995)
12. Oh, S.K., Pedrycz, W., Park, B.J.: Relation-based Neurofuzzy Networks with Evolutionary Data Granulation, Mathematical and Computer Modeling. (2003)
13. Oh, S.K., Pedrycz, W.: Fuzzy Polynomial Neuron-Based Self-Organizing Neural Networks, Int. J. of General Systems. 32 (3) (2003) 237-250
14. Park, B.J., Lee, D.Y., Oh, S.K.: Rule-based Fuzzy Polynomial Neural Networks in Modeling Software Process Data. Int. J. of Control Automation and Systems. 1(3) (2003) 321-331

Pricing the Foreign Currency Options with the Fuzzy Numbers Based on the Garman-Kohlhagen Model

Fan-Yong Liu

Financial Engineering Research Center, South China University of Technology,
Wushan, Guangzhou, P.R. China
fanyongliu@163.com

Abstract. This paper starts from the fuzzy environments of foreign currency options markets, introduces fuzzy sets theory, and gives a fuzzy version of Garman-Kohlhagen currency options pricing model. By taking exchange rate, domestic interest rate, foreign interest rate, and volatility as triangular fuzzy numbers, the currency option price will turn into a fuzzy number. This makes the financial investors who can pick any currency option price with an acceptable belief degree for the later use. In order to obtain the belief degree, an optimization procedure has been applied. An empirical study is performed based on market data. The study result indicates the fuzzy currency options pricing method is a useful tool for modeling the imprecise problem in the real world.

1 Introduction

Since the closed-form solution of the foreign currency options pricing model was derived by Garman and Kohlhagen [1] in 1983, many methodologies have been proposed by using the modification of Garman-Kohlhagen (G-K) model.

The input variables of the G-K model are usually regarded as the precise real numbers. However, these variables cannot always be expected in a precise sense. Therefore, the fuzzy sets theory proposed by Zadeh [2] may be a useful tool for modeling this kind of imprecise problem. The book [3] of collected papers gave the applications of using fuzzy sets theory to the financial engineering.

Next the motivation of this study is provided by explaining why it is need to take into account fuzzy variables in the G-K model, such as the fuzzy exchange rate, the fuzzy domestic interest rate, the fuzzy foreign interest rate, and the fuzzy volatility. In the constant interest rates approach, when the financial investor tries to price a currency option, the interest rates, both domestic and foreign, are assumed as constant. However, the interest rates may have the different values in the different commercial banks and financial institutions. Therefore, the choice of a reasonable interest rate may cause a dilemma. But one thing can be sure is that the different interest rates may be around a fixed value within a short period of time. For instance, the interest rate may be around 5%, the interest rate may be regarded as a fuzzy number 5% when the financial investor tries to price a currency option using the G-K model.

On the other hand, the financial market fluctuates from time to time. It is a little unreasonable to pick a fixed volatility to price a currency option at this time and then use this price for the later use, since the later volatility has changed. In this case, it is natural to regard the volatility as an imprecise data.

The usual approach for pricing is to pick fixed input data to price a currency option by applying the G-K model. However, this currency option price will be used for the further decision-making by a financial investor within a short period of time. The problem is that the input data will be changed within this short period of time because the international financial market fluctuates very irregular. Therefore, it is a little unreasonable to pick fixed input data to price a currency option at this time and then use this price for later use, since the later these input data has already changed. Therefore, it is also natural to assume these input data as fuzzy numbers.

Although it has been described that how the four fuzzy input variables occur in the real world, some of four variables can still be taken as the real (crisp) numbers if the financial investor can make sure that those variables occur in a crisp sense. In this case, the fuzzy model proposed in this paper is still applicable since the real numbers are the special case of the fuzzy numbers. Now, under the considerations of the four fuzzy variables, the currency option price will turn into a fuzzy number. This fuzzy number is in fact a function defined on \mathbb{R} into $[0, 1]$, and denoted by membership function $\mu_{\tilde{a}} : X \rightarrow [0, 1]$. Given any value c , the function value $\mu_{\tilde{a}}(c)$ will be interpreted as the belief degree of closeness to value a . It means that the closer the value c to a is, the higher the belief degree is. Therefore, the financial investors can pick any value that is around a with an acceptable belief degree as the currency option price for their later use. In order to obtain the belief degree of any a given option price, an optimization problem will be given. An efficient computational procedure [4] is applied in this paper to solve this optimization problem.

This paper is organized as follows. In Section 2, the notion and arithmetic of fuzzy numbers are introduced. In Section 3, the fuzzy version of the G-K model is given. In Section 4, the computational procedure is introduced in order to obtain the belief degrees of given option prices. In Section 5, the empirical study is performed. Finally, the conclusions and limitations are depicted.

2 Fuzzy Numbers

Let X be a universal set and A be a subset of X . The characteristic function is defined as $\chi_A : X \rightarrow \{0, 1\}$ on A . The function $\chi_A(a) = 1$ if $a \in A$, and $\chi_A(a) = 0$ if $a \notin A$. Zadeh [2] introduced the fuzzy subset \tilde{A} of X by extending the characteristic function $\chi_A : X \rightarrow \{0, 1\}$. A fuzzy subset \tilde{A} of X is defined by its membership function $\mu_{\tilde{A}} : X \rightarrow [0, 1]$. The value $\mu_{\tilde{A}}(a)$ can be interpreted as the membership degree (i.e. belief degree) of the point a in the set A .

Let X be a universal set and \tilde{A} be a fuzzy subset of X . The α -level set of \tilde{A} is defined by $\tilde{A}_\alpha = \{x | \mu_{\tilde{A}}(x) \geq \alpha\}$. The fuzzy subset \tilde{A} is called a normal fuzzy set if there exists x such that $\mu_{\tilde{A}}(x) = 1$. The fuzzy subset \tilde{A} is called a

convex fuzzy set if $\mu_{\tilde{A}}(\lambda x + (1 - \lambda)y) \geq \min\{\mu_{\tilde{A}}(x), \mu_{\tilde{A}}(y)\}$ for $\forall \lambda \in [0, 1]$. The universal set X is assumed to be a real number system; that is, $X = \mathbb{R}$. Let f be a real-valued function defined on \mathbb{R} . The function f is said to be upper semi-continuous, if $\{x|f(x) \geq \alpha\}$ is a closed set for each α . Or equivalently, f is upper semi-continuous at y if and only if $\forall \varepsilon > 0, \exists \delta > 0$ such that $|x - y| < \delta$ implies $f(x) < f(y) + \varepsilon$.

Under some suitable conditions for the membership function, the fuzzy set is then termed as a fuzzy number. The fuzzy number \tilde{a} corresponding to a can be interpreted as "around a ." The graph of the membership function $\mu_{\tilde{a}}(x)$ is bell shaped and $\mu_{\tilde{a}}(a) = 1$. It means that the membership degree $\mu_{\tilde{a}}(x)$ is close to 1 when the value x is close to a . \tilde{a} is called a fuzzy number if the following three conditions are satisfied: (i) \tilde{a} is a normal and convex fuzzy set; (ii) Its membership function $\mu_{\tilde{a}}(x)$ is upper semi-continuous; (iii) The α -level set \tilde{a}_α is bounded for each $\alpha \in [0, 1]$.

From Zadeh [2], \tilde{A} is a convex fuzzy set if and only if $\tilde{A}_\alpha = \{x|\mu_{\tilde{A}}(x) \geq \alpha\}$ is a convex set for all $\alpha \in [0, 1]$. Therefore, if \tilde{a} is a fuzzy number, then the α -level set \tilde{a}_α is a compact (closed and bounded in \mathbb{R}) and convex set; that is, \tilde{a}_α is a closed interval. The α -level set of \tilde{a} is then denoted by $\tilde{a}_\alpha = [\tilde{a}_\alpha^L, \tilde{a}_\alpha^U]$.

A fuzzy number \tilde{a} is said to be nonnegative if $\mu_{\tilde{a}}(x) = 0$ for $\forall x < 0$. It is easy to see that if \tilde{a} is a nonnegative fuzzy numbers then \tilde{a}_α^L and \tilde{a}_α^U are all nonnegative real numbers for all $\alpha \in [0, 1]$. The following proposition is useful for further discussion.

Proposition 1. (Resolution identity, Zadeh [5]). *Let \tilde{A} be a fuzzy set with membership function $\mu_{\tilde{A}}$ and $\tilde{A}_\alpha = \{x|\mu_{\tilde{A}}(x) \geq \alpha\}$. Then*

$$\mu_{\tilde{A}}(x) = \sup_{\alpha \in [0,1]} \alpha [1_{\tilde{A}_\alpha}(x)], \tag{1}$$

where $1_{\tilde{A}_\alpha}$ is an indicator function of set \tilde{A}_α , i.e., $1_{\tilde{A}_\alpha}(x) = 1$ if $x \in \tilde{A}_\alpha$ and $1_{\tilde{A}_\alpha}(x) = 0$ if $x \notin \tilde{A}_\alpha$. Note that the α -level set \tilde{A}_α of \tilde{A} is a crisp (usual) set. The \tilde{a} is called a crisp number with value m if its membership function is

$$\mu_{\tilde{a}}(x) = \begin{cases} 1, & \text{if } x = m, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The real numbers are the special case of the fuzzy numbers when the real numbers are regarded as the crisp numbers.

Now the algorithms of any two fuzzy numbers are introduced [6,7]. Let " \odot " be a binary operation \oplus, \ominus, \otimes , or \oslash between two fuzzy numbers \tilde{a} and \tilde{b} . The membership function of $\tilde{a} \odot \tilde{b}$ is defined by

$$\mu_{\tilde{a} \odot \tilde{b}}(z) = \sup_{\{(x,y)|x \odot y = z\}} \min\{\mu_{\tilde{a}}(x), \mu_{\tilde{b}}(y)\} \tag{3}$$

where the binary operations $\odot = \oplus, \ominus, \otimes$, or \oslash correspond to the binary operations $\circ = +, -, \times$, or $/$ according to the "Extension Principle" in Zadeh [5]. Let " \odot_{int} " be a binary operation $\oplus_{int}, \ominus_{int}, \otimes_{int}$, or \oslash_{int} between two closed intervals $[a, b]$ and $[c, d]$.

Proposition 2. *Let \tilde{a} and \tilde{b} be two fuzzy numbers. Then $\tilde{a} \oplus \tilde{b}$, $\tilde{a} \ominus \tilde{b}$ and $\tilde{a} \otimes \tilde{b}$ are also fuzzy numbers and their α -level sets are*

$$(\tilde{a} \oplus \tilde{b})_\alpha = \tilde{a}_\alpha \oplus_{int} \tilde{b}_\alpha = [\tilde{a}_\alpha^L + \tilde{b}_\alpha^L, \tilde{a}_\alpha^U + \tilde{b}_\alpha^U] \tag{4}$$

$$(\tilde{a} \ominus \tilde{b})_\alpha = \tilde{a}_\alpha \ominus_{int} \tilde{b}_\alpha = [\tilde{a}_\alpha^L - \tilde{b}_\alpha^U, \tilde{a}_\alpha^U - \tilde{b}_\alpha^L] \tag{5}$$

$$(\tilde{a} \otimes \tilde{b})_\alpha = \tilde{a}_\alpha \otimes_{int} \tilde{b}_\alpha = [\min\{\tilde{a}_\alpha^L \tilde{b}_\alpha^L, \tilde{a}_\alpha^L \tilde{b}_\alpha^U, \tilde{a}_\alpha^U \tilde{b}_\alpha^L, \tilde{a}_\alpha^U \tilde{b}_\alpha^U\}, \max\{\tilde{a}_\alpha^L \tilde{b}_\alpha^L, \tilde{a}_\alpha^L \tilde{b}_\alpha^U, \tilde{a}_\alpha^U \tilde{b}_\alpha^L, \tilde{a}_\alpha^U \tilde{b}_\alpha^U\}] \tag{6}$$

for all $\alpha \in [0, 1]$. If the α -level set \tilde{b}_α of \tilde{b} does not contain zero for all $\alpha \in [0, 1]$, then $\tilde{a} \oslash \tilde{b}$ is also a fuzzy number and its α -level set is

$$(\tilde{a} \oslash \tilde{b})_\alpha = \tilde{a}_\alpha \oslash_{int} \tilde{b}_\alpha = [\min\{\tilde{a}_\alpha^L / \tilde{b}_\alpha^L, \tilde{a}_\alpha^L / \tilde{b}_\alpha^U, \tilde{a}_\alpha^U / \tilde{b}_\alpha^L, \tilde{a}_\alpha^U / \tilde{b}_\alpha^U\}, \max\{\tilde{a}_\alpha^L / \tilde{b}_\alpha^L, \tilde{a}_\alpha^L / \tilde{b}_\alpha^U, \tilde{a}_\alpha^U / \tilde{b}_\alpha^L, \tilde{a}_\alpha^U / \tilde{b}_\alpha^U\}]. \tag{7}$$

Let \mathcal{F} denote the set of all fuzzy subsets of \mathbb{R} . Let $f(x)$ be a real-valued function from \mathbb{R} into \mathbb{R} and \tilde{A} be a fuzzy subset of \mathbb{R} . By the extension principle, the fuzzy-valued function $\tilde{f} : \mathcal{F} \rightarrow \mathcal{F}$ can be induced by the non-fuzzy $f(x)$; that is, $\tilde{f}(\tilde{A})$ is a fuzzy subset of \mathbb{R} . The membership function of $\tilde{f}(\tilde{A})$ is defined by

$$\mu_{\tilde{f}(\tilde{A})}(r) = \sup_{\{x|r=f(x)\}} \mu_{\tilde{A}}(x). \tag{8}$$

The following proposition is useful to discuss the fuzzy G-K model.

Proposition 3. *Let $f(x)$ be a real-valued function and \tilde{A} be a fuzzy subset of \mathbb{R} . The function $f(x)$ can induce a fuzzy-valued function $\tilde{f} : \mathcal{F} \rightarrow \mathcal{F}$ via the extension principle. Suppose that the membership function $\mu_{\tilde{A}}$ of \tilde{A} is upper semi-continuous and $\{x|r = f(x)\}$ is a compact set (it will be a closed and bounded set in \mathbb{R}) for all r , then the α -level set of $\tilde{f}(\tilde{A})$ is $(\tilde{f}(\tilde{A}))_\alpha = \{f(x)|x \in \tilde{A}_\alpha\}$.*

3 Fuzzy Version of Garman-Kohlhagen Model

The G-K model [1] for a European call currency option with expiry date T and strike price K is described as follows. Let S_t denotes the spot exchange rate at time $t \in [0, T]$. Let C_t denotes the price of this currency option at time t ,

$$C_t = S_t e^{-r_F \tau} N(d_1) - K e^{-r_D \tau} N(d_2), \quad \tau = (T - t) \tag{9}$$

$$d_1 = \frac{\ln(S_t/K) + (r_D - r_F + \sigma^2/2)\tau}{\sigma\sqrt{\tau}}, \quad d_2 = d_1 - \sigma\sqrt{\tau}$$

where r_F denotes the foreign interest rate, r_D denotes the domestic interest rate, σ denotes the volatility, and N stands for the cumulative distribution function of a standard normal random variable $N(0, 1)$.

Under the considerations of fuzzy exchange rate \tilde{S}_t , fuzzy interest rates \tilde{r}_F and \tilde{r}_D , and fuzzy volatility $\tilde{\sigma}$, the currency option price at time t is a fuzzy number and is denoted as \tilde{C}_t . The strike price K and time to maturity τ are real numbers.

According to Proposition 1, the membership function of \tilde{C}_t is given by

$$\mu_{\tilde{C}_t}(c) = \sup_{\alpha \in [0,1]} \alpha [1_{(\tilde{C}_t)_\alpha}(c)], \tag{10}$$

where $(\tilde{C}_t)_\alpha$ is α -level set of the fuzzy price \tilde{C}_t at time t . This α -level set is a closed interval and its left-end point and right-end point are displayed as:

$$(\tilde{C}_t)_\alpha = [(\tilde{C}_t)_\alpha^L, (\tilde{C}_t)_\alpha^U]. \tag{11}$$

From Proposition 3, since the function $N(x)$ is increasing, the α -level set of $\tilde{N}(\tilde{d})$ is given by

$$(\tilde{N}(\tilde{d}))_\alpha = \{N(x)|x \in \tilde{d}_\alpha\} = \{N(x)|x \in [\tilde{d}_\alpha^L, \tilde{d}_\alpha^U]\} = [N(\tilde{d}_\alpha^L), N(\tilde{d}_\alpha^U)]. \tag{12}$$

Similarly, e^{-x} is a decreasing function and $\ln(x)$ is an increasing function. Then the left-end point $(\tilde{C}_t)_\alpha^L$ and right-end point $(\tilde{C}_t)_\alpha^U$ of the closed interval $(\tilde{C}_t)_\alpha$ will be displayed as follows by using Proposition 2, 3 and Equation (12):

$$(\tilde{C}_t)_\alpha^L = (\tilde{S}_t)_\alpha^L e^{-(\tilde{r}_F)_\alpha^U \tau} N((\tilde{d}_1)_\alpha^L) - K e^{-(\tilde{r}_D)_\alpha^L \tau} N((\tilde{d}_2)_\alpha^U) \tag{13}$$

$$(\tilde{C}_t)_\alpha^U = (\tilde{S}_t)_\alpha^U e^{-(\tilde{r}_F)_\alpha^L \tau} N((\tilde{d}_1)_\alpha^U) - K e^{-(\tilde{r}_D)_\alpha^U \tau} N((\tilde{d}_2)_\alpha^L) \tag{14}$$

where

$$(\tilde{d}_1)_\alpha^L = \frac{\ln((\tilde{S}_t)_\alpha^L / K) + ((\tilde{r}_D)_\alpha^L - (\tilde{r}_F)_\alpha^U + (\tilde{\sigma}_\alpha^L)^2 / 2)\tau}{(\tilde{\sigma}_\alpha^U)\sqrt{\tau}} \tag{15}$$

$$(\tilde{d}_1)_\alpha^U = \frac{\ln((\tilde{S}_t)_\alpha^U / K) + ((\tilde{r}_D)_\alpha^U - (\tilde{r}_F)_\alpha^L + (\tilde{\sigma}_\alpha^U)^2 / 2)\tau}{(\tilde{\sigma}_\alpha^L)\sqrt{\tau}} \tag{16}$$

$$(\tilde{d}_2)_\alpha^L = (\tilde{d}_1)_\alpha^L - (\tilde{\sigma}_\alpha^U)\sqrt{\tau}, \quad (\tilde{d}_2)_\alpha^U = (\tilde{d}_1)_\alpha^U - (\tilde{\sigma}_\alpha^L)\sqrt{\tau} \tag{17}$$

4 Computational Method and Triangular Fuzzy Number

Given a European call currency option price c of the fuzzy price \tilde{C}_t at time t , it is important to know its belief degree α . If the financial investors are acceptable with this belief degree, then it will be reasonable to take the value c as the currency option price at time t . In this case, the financial investors can accept the value c as the currency option price at time t with belief degree α .

The membership function of fuzzy price \tilde{C}_t of a currency option at time t is given in Equation (10). Therefore, given any currency option price c , its belief degree can be obtained by solving the following optimization problem [4]:

$$\begin{aligned} (OP1) \quad & \text{maximum} \quad \alpha \\ & \text{subject to:} \quad (\tilde{C}_t)_\alpha^L \leq c \leq (\tilde{C}_t)_\alpha^U \\ & \quad \quad \quad 0 \leq \alpha \leq 1. \end{aligned}$$

Since $g(\alpha) = (\tilde{C}_t)_\alpha^L$ is an increasing function of α and $h(\alpha) = (\tilde{C}_t)_\alpha^U$ is a decreasing function of α , the optimization problem (OP1) can be rewritten as

$$\begin{aligned}
 (OP2) \quad & \text{maximum} \quad \alpha \\
 & \text{subject to: } g(\alpha) \leq c \\
 & \quad \quad \quad h(\alpha) \geq c \\
 & \quad \quad \quad 0 \leq \alpha \leq 1.
 \end{aligned}$$

Since $g(\alpha) = (\tilde{C}_t)_\alpha^L \leq (\tilde{C}_t)_\alpha^U = h(\alpha)$, one of the constraints $g(\alpha) \leq c$ or $h(\alpha) \geq c$ can be discarded in the following ways:

(i) If $g(1) \leq c \leq h(1)$ then $\mu_{\tilde{C}_t}(c) = 1$.

(ii) If $c < g(1)$ then $h(\alpha) \geq c$ is redundant, since $h(\alpha) \geq h(1) \geq g(1) > c$ for all $\alpha \in [0, 1]$ using the fact that $h(\alpha)$ is decreasing and $g(\alpha) \leq h(\alpha)$ for all $\alpha \in [0, 1]$. Thus, the following relaxed optimization problem will be solved:

$$\begin{aligned}
 (OP3) \quad & \text{maximum} \quad \alpha \\
 & \text{subject to: } g(\alpha) \leq c \\
 & \quad \quad \quad 0 \leq \alpha \leq 1.
 \end{aligned}$$

(iii) If $c > h(1)$ then $g(\alpha) \leq c$ is redundant, since $g(\alpha) \leq g(1) \leq h(1) < c$ for all $\alpha \in [0, 1]$ using the fact that $g(\alpha)$ is increasing and $g(\alpha) \leq h(\alpha)$ for all $\alpha \in [0, 1]$. Thus, the following relaxed optimization problem will be solved:

$$\begin{aligned}
 (OP4) \quad & \text{maximum} \quad \alpha \\
 & \text{subject to: } h(\alpha) \geq c \\
 & \quad \quad \quad 0 \leq \alpha \leq 1.
 \end{aligned}$$

The optimization (OP3) can be solved using the following algorithm [4].

Step 1: Let ε be the tolerance and α_0 be the initial value. Set $\alpha \leftarrow \alpha_0$, $low \leftarrow 0$, $up \leftarrow 1$.

Step 2: Find $g(\alpha)$. If $g(\alpha) \leq c$ then go to Step 3 otherwise go to Step 4.

Step 3: If $c - g(\alpha) < \varepsilon$ then EXIT and the maximum is α , otherwise set $low \leftarrow \alpha$, $\alpha \leftarrow (low + up)/2$ and go to Step 2.

Step 4: Set $up \leftarrow \alpha$, $\alpha \leftarrow (low + up)/2$ and go to Step 2.

For problem (OP4), it is enough to consider the equivalent constraint $-h(\alpha) \leq -c$, since $h(\alpha)$ is decreasing and continuous, i.e., $-h(\alpha)$ is increasing and continuous. Thus, the above algorithm is still applicable for solving (OP4).

The triangular fuzzy numbers are applied to denote the fuzzy input variables in G-K model because of their desirable properties [8]. The membership function of a triangular fuzzy number \tilde{a} is defined by

$$\mu_{\tilde{a}}(x) = \begin{cases} (x - a_1)/(a_2 - a_1), & \text{if } a_1 \leq x \leq a_2, \\ (a_3 - x)/(a_3 - a_2), & \text{if } a_2 < x \leq a_3, \\ 0, & \text{otherwise.} \end{cases} \tag{18}$$

which is denoted by $\tilde{a} = (a_1; a_2; a_3)$. The triangular fuzzy number \tilde{a} can be expressed as "around a_2 " or "being approximately equal to a_2 ". The real number

a_2 is called the core value of \tilde{a} , and a_1 and a_3 are called the left- and right-end values of \tilde{a} , respectively. The α -level set (a closed interval) of \tilde{a} is then

$$\tilde{a}_\alpha = \{x | \mu_{\tilde{a}}(x) \geq \alpha\} = [(1 - \alpha)a_1 + \alpha a_2, (1 - \alpha)a_3 + \alpha a_2] \tag{19}$$

$$\tilde{a}_\alpha^L = (1 - \alpha)a_1 + \alpha a_2, \quad \tilde{a}_\alpha^U = (1 - \alpha)a_3 + \alpha a_2. \tag{20}$$

5 Empirical Study

In this section, the proposed fuzzy version of the G-K model is tested with the daily EUR/USD currency option price data. The EUR/USD currency option price data come from the Finance Times and the British Banker’s Association websites. The market data cover the period 3-16-2006 to 4-18-2006. The number of data patterns is 22. This section consists of two parts. In part one, the price of EUR/USD currency option is studied under fuzzy environment at a given time. In part two, the price time series of EUR/USD currency option is studied from 3-16-2006 to 4-18-2006.

First, in part one, a European call EUR/USD currency option pricing is studied on March 16th 2006. The spot exchange rate is around 1.215, the 3-month volatility is around 9%, the domestic 3-month interest rate is around 4.93%, the foreign 3-month interest rate is around 2.71%, and the strike price is 1.21 with 3 months to expiry. The market price of this currency option under above conditions is 0.0274 USD.

The four fuzzy input variables are assumed to triangular fuzzy numbers, and $\tilde{r}_F = (2.69\%; 2.71\%; 2.72\%)$, $\tilde{r}_D = (4.91\%; 4.93\%; 4.95\%)$, $\tilde{\sigma} = (7.2\%; 9.0\%; 10.8\%)$ and $\tilde{S}_t = (1.2138; 1.2150; 1.2162)$, respectively. The fuzzy price \tilde{C}_t of this currency option can be obtained based on the above fuzzy model. Table 1 gives the belief degrees $\mu_{\tilde{C}_t}(c)$ for the possible EUR/USD currency option prices c by solving the optimization problems (OP3) and (OP4) using the computational procedure proposed above. If this currency option price \$0.027 is taken, then its belief degree is 0.9892. Therefore, if financial investors are tolerable with this belief degree 0.9892, then they can take this price \$0.027 for their later use. If the price is taken as $c = \$0.027872$, its belief degree will be 1.00. In fact, if $S_t = 1.215$, $r_F = 2.71\%$, $r_D = 4.93\%$ and $\sigma = 9.0\%$, then this option price will be \$0.027872 by Equation (9). This situation matches the observation that $c = \$0.027872$ has belief degree 1.00.

The α -level closed interval of the fuzzy price is shortening as α -level is rising. Table 2 gives the α -level closed interval of the fuzzy price of this currency option with the different α value. For $\alpha = 0.95$, it means that the currency option price

Table 1. The belief degrees for different currency option prices on 2006/03/16

| | | | | | | | |
|-------------------------|--------|--------|--------|--------|--------|--------|--------|
| Option Prices c | 2.50 | 2.60 | 2.70 | 2.80 | 2.90 | 3.00 | 3.10 |
| Belief Degrees α | 0.9644 | 0.9768 | 0.9892 | 0.9984 | 0.9860 | 0.9736 | 0.9612 |

Table 2. The α -level closed interval of the fuzzy price with the different α value

| Belief Degrees α | 0.90 | 0.92 | 0.94 | 0.96 | 0.98 | 0.99 | 1.00 |
|------------------------------------|--------|--------|--------|--------|--------|--------|--------|
| Left-end $(\tilde{C}_t)_\alpha^L$ | 1.9798 | 2.1414 | 2.3028 | 2.4643 | 2.6257 | 2.7064 | 2.7872 |
| Right-end $(\tilde{C}_t)_\alpha^U$ | 3.5946 | 3.4330 | 3.2715 | 3.1100 | 2.9486 | 2.8679 | 2.7872 |

will lie in the closed interval [2.3836; 3.1907] with belief degree 0.95. If financial investors are comfortable with this belief degree 0.95, then they can pick any value from the interval [2.3836; 3.1907] as the option price for their later use.

In Fig 1, the left diagram shows that the α -level varies with the different option price from 2.0 to 3.6 by the span 0.01 and the right diagram shows that the closed interval varies with the α -level from 0 to 1 by the span 0.02.

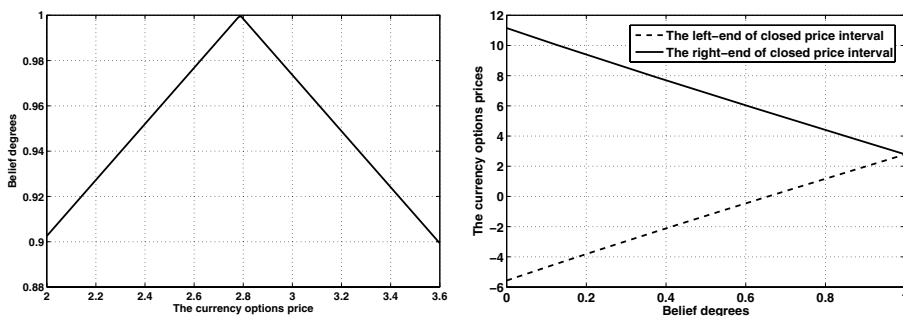


Fig. 1. The relation between the α -level and the currency option price

In part two, the experiment was performed based on the historical daily data of the EUR/USD currency option cover the period 3-16-2006 to 4-18-2006.

Fig 2 illustrates that the market prices lie in the closed interval with belief degree 90%. Therefore, if financial investors are tolerable with this belief degree 90%, then they can pick the daily market price or any price from this 90%-level interval as the option price for their later use at any trade date. In the Fig 2, the right- and left-end of the closed intervals with the different belief degrees are obtained by Equations (13) and (14). From Fig 2, most of market prices lie in the closed interval with belief degree 95% and thus the market prices are closer to the theory prices based on the G-K model with belief degree 1.

Fig 3 illustrates the maximum belief degrees of the currency option market prices to the theory price. The top diagram illustrates the market prices vary from 3-16-2006 to 4-18-2006. Correspondingly, the second diagram illustrates the belief degrees of the market prices. The belief degrees of most of the market prices are from 0.95 to 1 and the belief degrees of only a few market prices are between 0.90 and 0.95. Specially, only on April 7th 2006, the market price is 0.0278USD and its belief degree is less than 0.9. Therefore, the fuzzy version of the G-K model can simulate precisely the actual market prices in most cases.

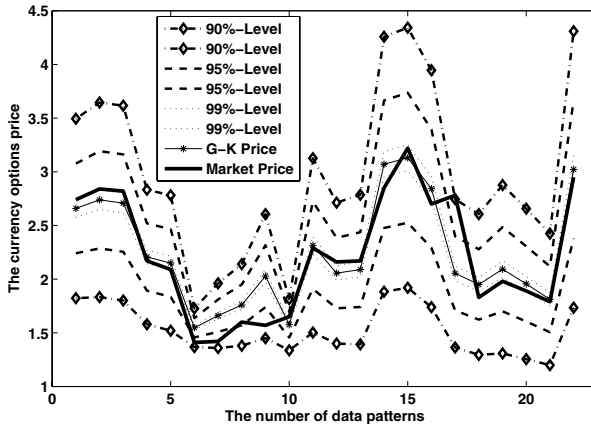


Fig. 2. The price interval with the different α -level (90%, 95%, 99%)

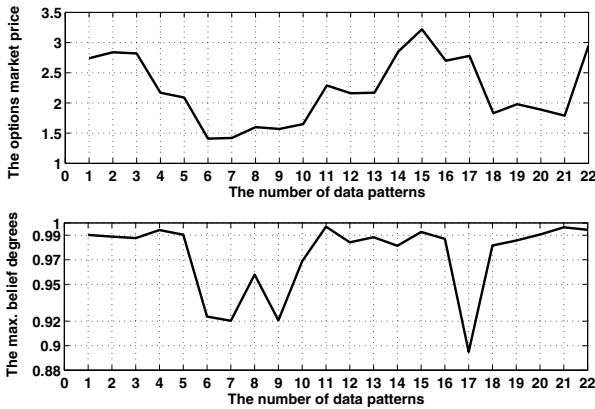


Fig. 3. The maximum belief degrees of the currency option market price series

6 Conclusions and Limitations

Owing to the fluctuation of international financial market from time to time, some input variables in the G-K model cannot always be expected in a precise sense. Therefore, the fuzzy version of G-K model based on fuzzy numbers is then proposed in this paper and the European call currency option price turns into a fuzzy number. This makes the financial investors who can pick any option price with an acceptable belief degree for their later use.

The proposed fuzzy G-K model is tested with the daily EUR/USD option market prices. The experimental results illustrate that the currency option market prices lie in the closed interval with belief degree 90% and most of market

prices lie in the closed interval with belief degree 95%. Thus, the fuzzy G-K model can simulate precisely the actual market prices in most cases.

In the usual approach, there are two ways to describe the input variable in the G-K model. One assumes the variable as a constant, and another one assumes the stochastic variable. The imprecisely stochastic input variable can be regarded as a fuzzy random variable. Only the fuzzy input variable is under investigated in this paper. In fact, it is still possible to take into account the fuzzy random variable for the options pricing. The study for the fuzzy random variable will be the future research.

According to Fig. 3, the belief degrees of a few market prices are between 0.90 and 0.95. Specially, on April 7th 2006, the belief degree of the market price is less than 0.9. Therefore, on one hand, the fuzzy G-K model can simulate the actual market prices in most cases. On the other hand, owing to the original G-K model is created under some assumptions, the fuzzy model based on G-K model may be imprecise when these assumptions are not true. The study for creating more precise fuzzy currency options pricing model will be future research.

References

1. Garman, M. B., Kohlhagen, S. W.: Foreign Currency Option Values. *Journal of International Money and Finance* **2** (1983) 231–237
2. Zadeh, L.A.: Fuzzy Sets. *Information and Control* **8(3)** (1965) 338–353
3. Ribeiro, R.A., Zimmermann, H.-J., Yager, R.R., Kacprzyk, J. (eds.): *Soft Computing in Financial Engineering*. Springer-Verlag, Berlin Heidelberg New York (1999)
4. Hsien-Chung W.: European Option Pricing under Fuzzy Environment. *International Journal of Intelligent System* **20** (2005) 89–102.
5. Zadeh, L.A.: The Concept of a Linguistic Variable and Its Application to Approximate Reasoning I. *Information Sciences* **8(3)** (1975) 199–251
6. Zadeh, L.A.: Fuzzy Algorithms. *Information and Control* **12** (1968) 94–102.
7. Santos, E.: Fuzzy Algorithms. *Information and Control* **17** (1970) 326–339
8. Andrés, J., Terceño, G.: Estimating a Term Structure of Interest Rates for Fuzzy financial Pricing by Using Fuzzy Regression Methods. *Fuzzy Sets and Systems* **139** (2003) 313–331

Designing Rough Sets Attributes Reduction Based Video Deinterlacing System

Gwanggil Jeon¹, Marco Anisetti², Valerio Bellandi², and Jechang Jeong¹

¹ Department of Electronics and Computer Engineering, Hanyang University,
17 Haengdang-dong, Seongdong-gu, Seoul, Korea
{windcap315, jjeong}@ece.hanyang.ac.kr

² Department of Information Technology, University of Milan,
via Bramante, 65 – 26013, Crema (CR), Italy
{anisetti, bellandi}@dti.unimi.it

Abstract. The use of rough set's theoretic concepts has permitted in this work to make the mathematical model on mode decision in deinterlacing system. In this paper, a rough set approach based decision making problem is proposed. In the literature, some conventional deinterlacing methods provide high performance with higher computational burden. On the other hand, some other methods give low performance with lower computational burden. Those all methods have been reported that interpolate missing pixels indiscriminately in the same way. Our algorithm chooses the most suitable method adaptively based on rough set theory using four parameters. This deinterlacing approach employs a size reduction of the database system, keeping only the essential information for the process, especially in the representation of and reasoning with vague and/or imprecise knowledge. Decision making and interpolation results are presented. The results of computer simulations show that the proposed method outperforms a number of methods presented in the literature.

1 Introduction

Recently, the several format based digital broadcast and progressive display system, like HDTV, requires the deinterlacing technology. Since the interlaced scanning process, such as NTSC, PAL, and SECAM, is applied in current television standards, the uncomfortable visual artifacts are produced. In order to solve above issue, many deinterlacing techniques have been proposed. Deinterlacing is a technique which converse interlaced scanning fields into progressive frames.

Deinterlacing methods can be roughly classified into three categories: spatial domain methods, which use only one field; temporal domain methods, which use multiple fields; and spatio-temporal domain methods [1-3]. The most common method in the spatial domain is Bob, which is used on small LCD panels [1]. However, the vertical resolution is halved, and this causes the image to have jagged edges. Weave is the most common method in the temporal domain [1]. However, this method gives motion artifacts. There exist many edge direction based interpolation methods. Oh *et al.* propose a spatio-temporal line average (STELA) algorithm [2]. STELA was proposed in order to expand the window to include the temporal domain.

Jeon and Jeong proposed fuzzy rule based direction oriented interpolation [FDOI] algorithm [3]. A line interpolation method that uses an intra- and inter-field edge-direction detector was proposed to obtain the correct edge information. The proposed rough set based deinterlacing algorithm employs four deinterlacing methods: Bob, Weave, STELA, and FDOI.

Generally, various features offer several attributes for the nature of a sequence. However, sometimes the attributes become too much to make essential rules. Although some rules are decided, even human experts are unable to believe the rules. Thus, the conventional deinterlacing method cannot be applied to build an expert system. In order to create an expert system, rough set theory is applied to classify the deinterlacing method [4]. In this theory, prior knowledge of the rules is not required, but rather the rules are automatically discovered from a database. Rough set theory has many interesting applications. It is turned out to be methodologically significant to artificial intelligence and cognitive science, especially in the representation of and reasoning with vague and/or imprecise knowledge, machine learning, knowledge acquisition, decision analysis, knowledge discovery from databases, expert systems and pattern recognition [5-14]. It seems of particular importance to decision support systems and data mining. Sugihara and Tanaka proposed a new rough set approach which deals with ambiguous and imprecise decision [15]. The deinterlacing technique causes a mode decision problem, because the mode decision method may affect interpolation efficiency, complexity, and objective and subjective results. We propose the study involving deinterlacing systems that are based on Sugihara's rough set theory.

This paper presents a decision making algorithm that is based on rough set theory for video deinterlacing. The operation of a decision in the deinterlacing method is intrinsically complex due to the high degree of uncertainty and the large number of variables involved. In Section 2, the background and definition of information system with ambiguous decision in evaluation problems are discussed. In Section 3, both of conventional deinterlacing methods and the proposed rough set based deinterlacing method are described. In Section 4, the experimental results and performance analysis are provided to show the feasibility of the proposed approach. These results are compared to well-known, pre-existing deinterlacing methods. Finally, conclusions are presented in Section 5.

2 Background and Definition of Information System with Ambiguous Decisions in Evaluation Problems

In the conventional rough set theory, it is assumed that the given values with respect to a decision attribute are certain. That is, each object x has only one decision value in the set of decision values. However, there exist some cases in which this assumption is not appropriate to real decision making problems. Sugihara and Tanaka considered the situations that decision values $d(x)$ are given to each object x as interval values [15].

Let Cl_n ($n=1, \dots, N$), be the n -th class with respect to a decision attribute. It is supposed that for all s, t , such that $t > s$, each element of Cl_t is preferred to each element of Cl_s . The interval decision classes (values) $Cl_{[s,t]}$ are defined as

$$Cl_{[s,t]} = \bigcup_{s \leq r \leq t} Cl_{[r]} \tag{1}$$

It is assumed that the decision of each $x \in U$ belongs to one or more classes, that is, $d(x) = Cl_{[s,t]}$. By $Cl_{[s,t]}$, a decision maker expresses ambiguous judgments to each object x . Based on the above equations, the decisions $d(x)$ with respect to the attribute set P can be obtained by the lower and upper approximations as follows.

The lower bounds $\underline{P}\{d(x)\}$ and the upper bounds $\overline{P}\{d(x)\}$ of $d(x)$ are defined as

$$\underline{P}\{d(x)\} = \bigcap_{y \in R_P(x)} d(y) \tag{2}$$

$$\overline{P}\{d(x)\} = \bigcap_{\{d(z) \supseteq d(y), d(y) | y \in R_P(x)\}} d(z) \tag{3}$$

$\underline{P}\{d(x)\}$ means that x certainly belongs to common classes which are assigned to all the element of the equivalence classes $R_P(x)$. $\overline{P}\{d(x)\}$ means that x may belong to the classes which are assigned to each element of the equivalence classes $R_P(x)$, respectively. It is obvious that the following inclusion relation $\underline{P}\{d(x)\} \subseteq d(x) \subseteq \overline{P}\{d(x)\}$. Equations (2) and (3) are based on the concept of *greatest lower* and *least upper*, respectively.

3 Applied Example: Rough Set Based Deinterlacing System

3.1 Conventional Deinterlacing Methods

Bob is an intra field interpolation method which uses the current field to interpolate the missing field and to reconstruct one progressive frame at a time. Let $x(i, j-1, k)$ and $x(i, j+1, k)$ denote the upper reference line and the lower reference line, respectively. The current pixel $x_{Bob}(i, j, k)$ is then determined by:

$$x_{Bob}(i, j, k) = \{x(i, j-1, k) + x(i, j+1, k)\} \gg 1 \tag{4}$$

Inter-field deinterlacing is a simple deinterlacing method. The output frame $x_{Weave}(i, j, k)$ is defined as (5),

$$x_{Weave}(i, j, k) = \begin{cases} x(i, j, k), & j \bmod 2 = n \bmod 2 \\ x(i, j, k-1), & \text{otherwise} \end{cases} \tag{5}$$

where (i, j, k) designating the position, $x(i, j, k)$ the input field defined for $j \bmod 2 = n \bmod 2$ only, k is field number. It is well-known that the video quality of the inter-field interpolation is better than that of intra-field interpolation in a static area. However, the line-crawling effect occurs in areas of motion. The STELA algorithm performs the edge-based line averaging on the spatio-temporal window [3]. In order to alleviate the interpolation error caused by high horizontal frequency component, a directional-based interpolation method is applied to the low-pass filtered signal. The FDOI algorithm is based on line interpolation method that uses an

intra- and inter- field edge-direction detector was proposed to obtain the correct edge information.

RSD employs four deinterlacing methods: Bob, Weave, STELA, and FDOI. Bob exhibits no motion artifacts and has minimal computational requirements. And this method requires only 3 % of computational CPU time than that of FDOI. However, the input vertical resolution is halved before the image is interpolated, thus reducing the detail in the progressive image. And it causes staircase artifacts in edge region. Weave results no degradation in static images. However, the edges exhibit significant serrations, which is an unacceptable artifact in a broadcast or professional television environment. From Table 1, it can be seen that the processing requirements for Weave are almost same or slightly less than that of Bob method, 27.93% of STELA method, and 2.85% of FDOI method. STELA has merits in both of objective and subjective quality; moreover it prevents the staircase artifacts. However, STELA is useful on the edges with -45°, 0°, and 45° degrees and it cannot be a perfect solution in all time. Since STELA method is the extended one from Bob or Weave, STELA guarantees the performance of Bob or Weave. From Table 3, we can see that the performance of STELA is higher than that of Bob or Weave yet with the disadvantage of higher complexity. We select complex method as FDOI which requires lots of computational CPU time, but gives best objective and subjective performance. FDOI has about ten times higher complexity than STELA, yet it guarantees the best results.

3.2 Rough Set Based Deinterlacing Method

Many classification patterns exist for images. In this paper, it is assumed that an image can be classified by four main parameters: TD, SD, TMDW, and SMDW. The temporal difference (TD) or spatial difference (SD) is the pixel difference between two values across the missing pixel in each domain. The TMDW (or SMDW) parameter represents the temporal (or spatial) entropy. The characteristics of TMDW and SMDW are described in [3].

Table 1. Comparison of the normalized average CPU time among four deinterlacing methods with six above CIF test sequences

| Method | Sequences | | | | | | RT_i |
|--------|-----------|--------------|---------|--------|--------|--------|----------------------|
| | Akiyo | Hall Monitor | Foreman | News | Mobile | Stefan | Approximated average |
| Bob | 0.0302 | 0.0326 | 0.0300 | 0.0308 | 0.0262 | 0.0322 | 0.03 |
| Weave | 0.0268 | 0.0303 | 0.0294 | 0.0278 | 0.0259 | 0.0308 | 0.03 |
| STELA | 0.1020 | 0.1053 | 0.1027 | 0.1050 | 0.0924 | 0.1045 | 0.10 |
| FDOI | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.00 |

The first step of the proposed algorithm is to redefine the value of the each attribute, according to a certain metric. Using 100 frames (2nd to 101st frames) of each of the six CIF sequences (Akiyo, Hall Monitor, Foreman, News, Mobile, and Stefan) as the training data, the decision making map can be obtained through the training

process. Table 1 shows a comparison of the normalized average CPU time among the four methods. In case of TD and SD, the numerical range is linearly divided into two categories: S (small) and L (Large). In case of TMDW and SDMW, the numerical range is linearly divided into three categories: S (small), M (medium), and L (large).

Since each sequence has different degrees of spatial and temporal details, it is a difficult process to design consistent decision making tables. The detail required to determine $abcd/U$ is described in Table 2.

Table 2. Fuzzy rules for determination of attributes $a, b, c,$ and d

| | | | | |
|---|----|--|------|------------|
| 1 | IF | TD is smaller than 2^3 | THEN | a is S |
| | IF | TD is larger than 2^3 | THEN | a is L |
| 2 | IF | SD is smaller than 2^3 | THEN | b is S |
| | IF | SD is larger than 2^3 | THEN | b is L |
| 3 | IF | $TMDW$ is smaller than 2^2 | THEN | c is S |
| | IF | $TMDW$ is larger than 2^2 and smaller than 2^4 | THEN | c is M |
| | IF | $TMDW$ is larger than 2^4 | THEN | c is L |
| 4 | IF | $SMDW$ is smaller than 2^2 | THEN | d is S |
| | IF | $SMDW$ is larger than 2^2 and smaller than 2^4 | THEN | d is M |
| | IF | $SMDW$ is larger than 2^4 | THEN | d is L |

The set of all possible decisions are listed in Table 3, which collected through several training sequences. The proposed information system is composed of $R=[a,b,c,d,m|\{a,b,c,d\} \rightarrow \{m\}]$ as shown in Table 3. This table is a decision table in which $a, b, c,$ and d are condition attributes, whereas m is a decision attribute. Using these values, a set of examples can be generated. The attribute m represents selected method which is the decision maker’s decision: Bob assigned to B , Weave assigned to W , STELA assigned to T , and FDOI assigned to F .

It is assumed that the average absolute difference between the real value and the Bob method utilized the interpolated value as AD_B as shown in Table 3. In the same manner, $AD_W, AD_T,$ and AD_F are obtained.

Since each method has its own merits and demerits, RSD method is based on variable deinterlacing mode technique. And this technique causes a mode decision problem, because the mode decision method may affect interpolation efficiency, complexity, and objective and subjective results. As rate-distortion optimization (RDO) of reference software in H.264, we proposed a rule to select the suitable methods in each condition. This rule has been applied to various video sequences and supplies good performance in terms of PSNR and complexity. The goal of the rule is to select the mode which has minimum average cost in which given computational CPU time.

$$C_i = AD_i + \kappa \cdot RT_i \tag{6}$$

where $i \in \{B, W, T, F\}$, C_i is the cost in each method i , AD_i is the average absolute difference, RT_i is the expected required computational CPU time, and the parameter κ

Table 3. Set of the selected method corresponding to each pattern

| <i>abcd(U)</i> | <i>P</i> | <i>AD_B</i> | <i>AD_W</i> | <i>AD_T</i> | <i>AD_F</i> | <i>C_B</i> | <i>C_W</i> | <i>C_T</i> | <i>C_F</i> | <i>m</i> |
|----------------|----------|-----------------------|-----------------------|-----------------------|-----------------------|----------------------|----------------------|----------------------|----------------------|----------|
| SSSS | 35.14% | 1.33 | 1.64 | 1.15 | 1.06 | 2.05 | 2.36 | 2.65 | 4.06 | B |
| SSSM | 11.17% | 2.50 | 3.24 | 2.12 | 2.08 | 3.22 | 3.96 | 3.62 | 5.08 | B |
| SSSL | 0.97% | 6.12 | 6.19 | 4.25 | 4.15 | 6.84 | 6.91 | 5.75 | 7.15 | T |
| SSMS | 2.76% | 2.85 | 2.92 | 2.31 | 2.23 | 3.57 | 3.64 | 3.81 | 5.23 | B,W |
| SSMM | 5.37% | 3.09 | 3.83 | 2.74 | 2.72 | 3.81 | 4.55 | 4.24 | 5.72 | B |
| SSML | 1.59% | 5.66 | 6.41 | 4.11 | 4.07 | 6.38 | 7.13 | 5.61 | 7.07 | T |
| SSSL | 0.07% | 6.40 | 4.05 | 3.90 | 3.63 | 7.12 | 4.77 | 5.40 | 6.63 | W |
| SSLM | 0.35% | 7.06 | 5.27 | 4.69 | 4.47 | 7.78 | 5.99 | 6.19 | 7.47 | W |
| SLLL | 1.67% | 5.04 | 7.42 | 4.57 | 4.49 | 5.76 | 8.14 | 6.07 | 7.49 | B |
| SLSS | 3.60% | 7.64 | 4.39 | 3.72 | 3.58 | 8.36 | 5.11 | 5.22 | 6.58 | W |
| SLSM | 3.51% | 10.17 | 5.70 | 5.45 | 5.39 | 10.89 | 6.42 | 6.95 | 8.39 | W |
| SLSL | 1.45% | 14.46 | 9.46 | 7.97 | 7.84 | 15.18 | 10.18 | 9.47 | 10.84 | T |
| SLMS | 0.88% | 10.05 | 6.00 | 5.90 | 5.79 | 10.77 | 6.72 | 7.40 | 8.79 | W |
| SLMM | 2.44% | 9.38 | 6.80 | 6.86 | 6.47 | 10.10 | 7.52 | 8.36 | 9.47 | W |
| SLML | 1.88% | 12.57 | 10.04 | 7.23 | 6.83 | 13.29 | 10.76 | 8.73 | 9.83 | T |
| SLLS | 0.15% | 16.86 | 8.98 | 8.05 | 7.99 | 17.58 | 9.70 | 9.55 | 10.99 | T |
| SLLM | 0.60% | 15.39 | 8.58 | 7.91 | 7.91 | 16.11 | 9.30 | 9.41 | 10.91 | W |
| SLLL | 2.15% | 11.27 | 11.27 | 7.17 | 6.42 | 11.99 | 11.99 | 8.67 | 9.42 | T |
| LSSS | 1.50% | 4.88 | 9.76 | 4.14 | 3.38 | 5.60 | 10.48 | 5.64 | 6.38 | B,T |
| LSSM | 2.03% | 6.29 | 12.08 | 5.55 | 5.12 | 7.0 | 12.80 | 7.05 | 8.12 | B,T |
| LSSL | 0.58% | 7.25 | 25.26 | 7.02 | 6.62 | 7.97 | 25.98 | 8.52 | 9.62 | B |
| LSMS | 0.69% | 6.25 | 9.73 | 5.03 | 4.30 | 6.97 | 10.45 | 6.53 | 7.30 | T |
| LSMM | 1.90% | 6.48 | 10.82 | 5.74 | 5.62 | 7.20 | 11.54 | 7.24 | 8.62 | B,T |
| LSML | 0.92% | 7.31 | 17.40 | 6.63 | 6.07 | 8.03 | 18.12 | 8.13 | 9.07 | B |
| LLSL | 0.09% | 9.12 | 7.72 | 5.43 | 4.88 | 9.84 | 8.44 | 6.93 | 7.88 | T |
| LSLM | 0.36% | 9.11 | 10.87 | 6.80 | 6.20 | 9.83 | 11.59 | 8.30 | 9.20 | T |
| LSSL | 1.10% | 8.10 | 15.30 | 7.55 | 7.06 | 8.82 | 16.02 | 9.05 | 10.06 | B |
| LLSS | 1.15% | 14.44 | 19.41 | 9.89 | 8.34 | 15.16 | 20.13 | 11.39 | 11.34 | F,T |
| LLSM | 2.06% | 11.66 | 19.93 | 9.39 | 8.76 | 12.38 | 20.65 | 10.89 | 11.76 | T |
| LSSL | 1.34% | 12.48 | 28.10 | 11.37 | 10.62 | 13.20 | 28.82 | 12.87 | 13.62 | T |
| LLMS | 0.79% | 14.74 | 16.61 | 10.24 | 8.31 | 15.46 | 17.33 | 11.74 | 11.31 | F |
| LLMM | 2.48% | 13.38 | 18.67 | 10.34 | 8.87 | 14.10 | 19.39 | 11.84 | 11.87 | T,F |
| LLML | 2.49% | 13.10 | 23.06 | 10.94 | 10.60 | 13.82 | 23.78 | 12.44 | 13.60 | T |
| LLLS | 0.22% | 19.35 | 17.72 | 11.59 | 8.77 | 20.07 | 18.44 | 13.09 | 11.77 | F |
| LLLM | 1.01% | 18.91 | 16.38 | 11.93 | 11.43 | 19.63 | 17.10 | 13.43 | 14.43 | T |
| LLLL | 3.54% | 12.84 | 19.64 | 10.11 | 9.73 | 13.56 | 20.36 | 11.61 | 12.73 | T |
| average | 100.0% | 9.54 | 11.41 | 6.66 | 6.16 | | | | | |

is determined before experiment (simulation results presented for $\kappa=1.5$). It is assumed that the method which has the least cost is chosen as the selected method in each condition. However, it is difficult to choose the suitable method in some conditions, such as rules SSMS, LSSS, LSSM, LSMM, LLSS, and LLMM), because the difference of cost between two best methods is too small (less than 0.1).

Table 4 shows the extracted 42 evaluation rules in deinterlacing system. The system designer assigns the suitable methods: Bob, Weave, STELA, and FDOI.

Table 4. The information system (evaluation rules)

| Staff | <i>a</i> | <i>b</i> | <i>c</i> | <i>d</i> | <i>m</i> | Staff | <i>a</i> | <i>b</i> | <i>c</i> | <i>d</i> | <i>m</i> |
|-------|----------|----------|----------|----------|----------|-------|----------|----------|----------|----------|----------|
| 1 | S | S | S | S | B | 22 | L | S | S | M | B |
| 2 | S | S | S | M | B | 23 | L | S | S | M | T |
| 3 | S | S | S | L | T | 24 | L | S | S | L | B |
| 4 | S | S | M | S | B | 25 | L | S | M | S | T |
| 5 | S | S | M | S | W | 26 | L | S | M | M | B |
| 6 | S | S | M | M | B | 27 | L | S | M | M | T |
| 7 | S | S | M | L | T | 28 | L | S | M | L | B |
| 8 | S | S | L | S | W | 29 | L | S | L | S | T |
| 9 | S | S | L | M | W | 30 | L | S | L | M | T |
| 10 | S | S | L | L | B | 31 | L | S | L | L | B |
| 11 | S | L | S | S | W | 32 | L | L | S | S | F |
| 12 | S | L | S | M | W | 33 | L | L | S | S | T |
| 13 | S | L | S | L | T | 34 | L | L | S | M | T |
| 14 | S | L | M | S | W | 35 | L | L | S | L | T |
| 15 | S | L | M | M | W | 36 | L | L | M | S | F |
| 16 | S | L | M | L | T | 37 | L | L | M | M | T |
| 17 | S | L | L | S | T | 38 | L | L | M | M | F |
| 18 | S | L | L | M | W | 39 | L | L | M | L | T |
| 19 | S | L | L | L | T | 40 | L | L | L | S | F |
| 20 | L | S | S | S | B | 41 | L | L | L | M | T |
| 21 | L | S | S | S | T | 42 | L | L | L | L | T |

where

$$\begin{aligned}
 U &= \{1, 2, 3, \dots, 40, 41, 42\} \\
 C &= \{a \text{ (TD)}, b \text{ (SD)}, c \text{ (TMDW)}, d \text{ (SMDW)}\} \\
 V_{TD} &= V_{SD} = \{S, L\} \\
 V_{TMDW} &= V_{SMDW} = \{S, M, L\} \\
 \{d\} &= \{B, W, T, F\}
 \end{aligned}$$

From the indiscernibility relations, the lower bounds of the decisions $d(x)$ and the upper bounds of the decisions $d(x)$ for each object x are obtained as follows:

$$\begin{aligned}
 \underline{P}\{d(4)\} &= \phi, \bar{P}\{d(4)\} = ["B, W"] & \underline{P}\{d(26)\} &= \phi, \bar{P}\{d(26)\} = ["B, T"] \\
 \underline{P}\{d(5)\} &= \phi, \bar{P}\{d(5)\} = ["B, W"] & \underline{P}\{d(27)\} &= \phi, \bar{P}\{d(27)\} = ["B, T"] \\
 \underline{P}\{d(20)\} &= \phi, \bar{P}\{d(20)\} = ["B, T"] & \underline{P}\{d(32)\} &= \phi, \bar{P}\{d(32)\} = ["F, T"] \\
 \underline{P}\{d(21)\} &= \phi, \bar{P}\{d(21)\} = ["B, T"] & \underline{P}\{d(33)\} &= \phi, \bar{P}\{d(33)\} = ["F, T"] \\
 \underline{P}\{d(22)\} &= \phi, \bar{P}\{d(22)\} = ["B, T"] & \underline{P}\{d(37)\} &= \phi, \bar{P}\{d(37)\} = ["T, F"] \\
 \underline{P}\{d(23)\} &= \phi, \bar{P}\{d(23)\} = ["B, T"] & \underline{P}\{d(38)\} &= \phi, \bar{P}\{d(38)\} = ["T, F"]
 \end{aligned} \tag{7}$$

If $f(x, q_{TD}) = "S," f(x, q_{SD}) = "S," f(x, q_{TMDW}) = "M,"$ and $f(x, q_{SMDW}) = "S"$ then exactly \emptyset .
 (Supported by 4, 5)

If $f(x, q_{TD}) = "S," f(x, q_{SD}) = "S," f(x, q_{TMDW}) = "M,"$ and $f(x, q_{SMDW}) = "S"$ then possibly $d(x) = ["B, W"]$.
 (Supported by 4, 5)

The other rules have crisp decision, e.g.,

$$\begin{aligned}
 \underline{P}\{d(1)\} = \overline{P}\{d(1)\} &= ["B"] & \dots \\
 \underline{P}\{d(2)\} = \overline{P}\{d(2)\} &= ["B"] & \underline{P}\{d(40)\} = \overline{P}\{d(40)\} &= ["F"] \\
 \underline{P}\{d(3)\} = \overline{P}\{d(3)\} &= ["T"] & \underline{P}\{d(41)\} = \overline{P}\{d(41)\} &= ["T"] \\
 & \dots & \underline{P}\{d(42)\} = \overline{P}\{d(42)\} &= ["T"]
 \end{aligned}
 \tag{8}$$

The priority of rule 4 and rule 5 are the same, thus we can use any method in those condition (in our simulation, priority order is given *F*, *T*, *W* and *B*). The final results, presented in Table 4, can be rewritten as a minimal decision algorithm in normal form which based on original rough set theory [4]. Combining the decision rules into one decision class provides the following decision algorithm.

$$\begin{aligned}
 & \text{if } (a_L b_L (d_S \vee c_M d_M)) \quad m_F \\
 & \text{else if } (b_S ((c_S (d_S \vee d_M)) \vee c_M d_M \vee c_L d_L) \vee a_L d_L) \quad m_B \\
 & \text{else if } ((d_S (a_S b_S c_L \vee b_L c_S \vee a_S c_M)) \vee (d_M (a_S c_L \vee a_S b_L \vee b_L c_M))) \quad m_W \\
 & \text{else } \quad m_T
 \end{aligned}
 \tag{9}$$

4 Experimental Results

In this section, we compare the subjective, objective quality and computational CPU time of our proposed method with other methods. The proposed algorithm was implemented on a Pentium IV/3.20 GHz computer. For the objective and subjective performance evaluation, four 1920*1080i HDTV video sequences were selected to challenge the five algorithms for Bob, Weave, STELA, FDOI, and the proposed method. Table 5 shows the comparison result of different deinterlacing methods for various sequences. The results show that the proposed method demonstrates the 2nd best objective performance compared to the other conventional methods, in terms of PSNR.

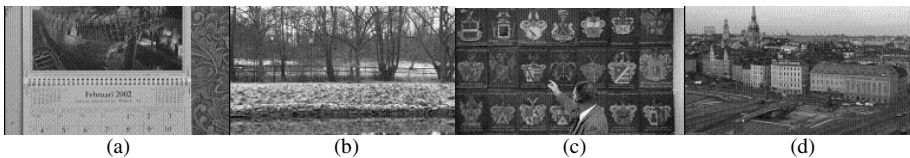


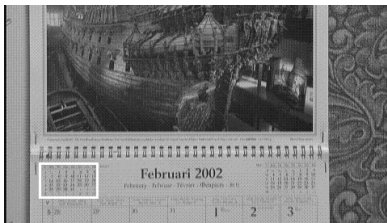
Fig. 1. Four 1920*1080i test sequences that are used: (a) Mobcal, (b) Parkrun, (c) Shields, and (d) Stockholm

Fig. 2 shows the subjective quality by the methods mentioned above. The superior performance is obvious especially at the outline of the characters. Mobcal sequence has following characteristics; Mobcal sequence has high spatial detail and medium amount of motion. The camera pans vertically (top to bottom). Because the sequence has a lot of motion, the information comes from inter fields is not good enough to be used. Therefore, Bob method provides almost the same performance with FDOI. Our

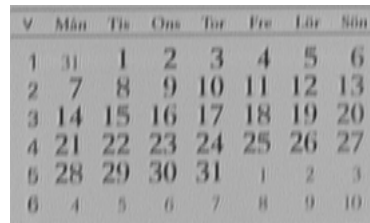
proposed algorithm gets more pleasing visual quality and significantly improves edge flicker, interline flicker, and line crawling by accurately assigning the most suitable deinterlacing methods. Computational CPU time for various methods is shown in Table 5 as well.

Table 5. Results of different deinterlacing methods for four interlaced HDTV sequences

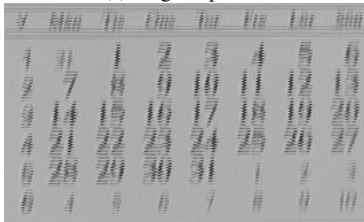
| Method | Sequences (PSNR, computational CPU time) (Unit = dB, ms) | | | | | | | |
|----------|--|--------|---------|--------|---------|--------|-----------|--------|
| | Mobcal | | Parkrun | | Shields | | Stockholm | |
| Bob | 32.2072 | 0.140 | 21.2252 | 0.187 | 24.1377 | 0.172 | 26.4087 | 0.125 |
| Weave | 26.8416 | 0.125 | 19.1426 | 0.172 | 21.4058 | 0.125 | 24.0388 | 0.140 |
| STELA | 30.4403 | 0.672 | 21.3727 | 0.547 | 24.2633 | 0.563 | 26.5745 | 0.625 |
| FDOI | 32.5208 | 15.383 | 21.5598 | 14.937 | 24.5898 | 14.719 | 26.7630 | 15.680 |
| Proposed | 32.3435 | 0.843 | 21.3927 | 0.765 | 24.2717 | 0.750 | 26.5808 | 0.937 |



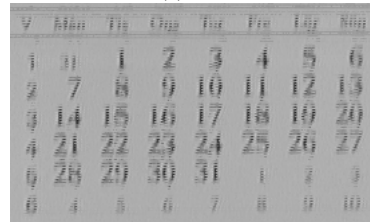
(a) Original picture



(b) Bob



(c) Weave



(d) STELA



(e) FDOI



(f) Proposed method

Fig. 2. Subjective quality comparison of the 109th Mobcal 1920*1080i HDTV sequence

5 Conclusion

This paper describes an application of rough set to decision making problems in deinterlacing. In decision making problems, there are several cases where the decision maker’s judgments are uncertainty. Using rough set theory, the deinterlacing system with ambiguous decisions which given by a decision maker is dealt with. The

different method from the conventional rough set analysis is illustrated on the assumption that the decision values are comparable. Decision making and interpolation results are presented. The results of computer simulations show that the proposed method outperforms a number of methods in literature.

Acknowledgment. “This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (KRF-2006-005-J04101) and Brain Korea 21 Project 2007”.

References

1. M. -J. Chen, C. -H. Huang, and C. -T. Hsu, “Efficient de-interlacing technique by inter-field information,” *IEEE Trans. Consumer Electronics*, vol. 50, no. 4, pp. 1202-1208, Nov. 2004
2. H. -S. Oh, Y. Kim, Y. -Y. Jung, A. W. Morales, and S. -J. Ko, “Spatio-temporal edge-based median filtering for deinterlacing,” *IEEE International Conference on Consumer Electronics*, pp. 52-53, 2000
3. G. Jeon and J. Jeong, “Designing Takagi-Sugeno fuzzy model-based motion adaptive deinterlacing system,” *IEEE Trans. Consumer Electronics*, vol. 52, no. 3, pp. 1013-1020, Aug. 2006
4. Z. Pawlak - “Rough Sets - Theoretical Aspects of Reasoning about Data,” Kluwer Academic Publishers, 1991
5. X. -F. Zhang, F. -Z. Zhang, and Y. -S. Zhao, “Generalization of RST in ordered information table,” in *Proc. ICMLC 2005, Guangzhou, China, 2005*, pp. 2027 - 2032 Vol. 4
6. L. Pan, H. Zheng, S. Nahavandi, “The application of rough set and Kohonen network to feature selection for object extraction,” in *Proc. ICMLC 2003, Xi’an, China, 2003*, pp. 1185 - 1189 Vol.2
7. J. W. Grzymala-Busse, “LERS - A system for learning from examples based on rough sets,” in R. Slowinski (Ed.) *Intelligent Decision Support. Handbook of Applications and Advances of the Rough Set Theory*, Kluwer Academic Publishers, Dordrecht, 1992
8. Xiaohua Hu, “Using rough sets theory and database operations to construct a good ensemble of classifiers for data mining applications,” in *Proc. ICDM 2001, San Jose, California, 2001*, pp. 233 - 240.
9. X. Wu and Q. Wang “Application of rough set attributes reduction in quality evaluation of dissertation,” in *Proc. ICGC 2006, Atlanta, GA, 2006*, pp. 562 – 565
10. Y. Peng, G. Liu, T. Lin, and H. Geng, “Application of rough set theory in network fault diagnosis,” in *Proc. ICITA 2005, Hangzhou, China, 2005*, 556 - 559 vol.2
11. A. Kusiak, “Rough Set Theory: A data mining tool for semiconductor manufacturing,” *IEEE Trans. Electronics Packaging Manufacturing*, vol. 24, no. 1, pp. 44-50, Jan. 2001
12. A. K. Agrawal and A. Agarwal, “Rough logic for building a landmine classifier,” in *Proc. ICNSC 2005, Tucson, AZ, 2005*, pp. 855 – 860
13. F. Su, C. Zhou, and W. Shi, “Goevent association rule discovery model based on rough set with marine fishery application,” in *Proc. IGARSS 2004, Anchorage, Alaska, 2004*, pp. 1455 - 1458 vol.2.
14. L. Torres, “Application of rough sets in power system control center data mining,” in *Proc. PESW 2002, New York, NY, 2002*, pp. 627 - 631 vol.1.
15. K. Sugihara and H. Tanaka, “Rough set approach to information systems with interval decision values in evaluation problems,” in *Proc. ISFUROS 2006, Santa Clara, Cuba, 2006*.

Optimization of Fuzzy Membership Function Using Clonal Selection

Ayşe Merve Şakiroğlu and Ahmet Arslan

Selcuk University, Eng.-Arch. Fac. Computer Eng.42075-Konya, Turkey
{msakiroglu, ahmetarslan}@selcuk.edu.tr

Abstract. A clonal selection algorithm (Clonalg) inspires from Clonal Selection Principle used to explain the basic features of an adaptive immune response to an antigenic stimulus. It takes place in various scientific applications and it can be also used to determine the membership functions in a fuzzy system. The aim of the study is to adjust the shape of membership functions and a novice aspect of the study is to determine the membership functions. Proposed method has been implemented using a developed Clonalg program for a single input and output fuzzy system. In the previous work [1], using genetic algorithm (GA) is proposed to it. In this study they are compared, too and it has been shown that using clonal selection algorithm is advantageous than using GA for finding optimum values of fuzzy membership functions.

1 Introduction

Fuzzy logic is used to solve a lot of problems related wide range of area because of providing flexible solutions. Designing fuzzy system contains fuzzy sets which are defined by rule table and membership functions. When the fuzzy sets have been established, how best to determine the membership function is the first question that has to be tackled. For solving this problem, some methods such as genetic algorithms (GA), self-organizing feature maps (SOFM), tabu search (TS) etc. can be used.

GA was used by Karr [10] in determination of membership functions. Karr applies GA to design of fuzzy logic controller (FLC) for the cart pole problem. Meredith et al. [12] have applied GA to the fine tuning of membership functions in a FLC for a helicopter. Lee and Takagi [11] also tackle the cart problem. They take a holistic approach by using GA to design the whole system. Cheng et al. [2] have chosen the image threshold via minimizing the measure of fuzziness. For selecting the bandwidth of fuzzy membership functions, they use peak locations which are chosen from the histogram using the peak selection criterion. Bağış [3] presents a method for the determination of the membership functions based on the use of Tabu Search algorithm. Cerrada et al. [4] proposed an approach permits incorporate the temporal behavior of the system variables into the fuzzy membership functions. Simon [5] employed H_{∞} state estimation theory for the membership function parameter optimization. He made some modifications on the H_{∞} filter with addition of state constraints so that the resulting membership functions are sum normal. Yang and Bose [6] described a method

for generating a fuzzy membership functions with unsupervised learning using self-organizing feature map. They applied this method to pattern recognition.

In the previous work, proposed a new method for determination of fuzzy logic membership functions implemented using a genetic algorithm program for single input and output system. Due to space limitation in this paper, how the membership functions compute using GA didn't explained. Details of it can be found in the earlier work [1]. In this work the same method was performed using an artificial immune system algorithm: Clonalg. This algorithm is inspired from Clonal Selection Principle used to explain the basic features of an adaptive immune response to an antigenic stimulus. Both of these algorithms were coded and acquired results were compared.

This paper is organized as follow. In Sect. 2, basic principles and features of Clonalg are explained. In Sect. 3, how the membership functions that are in a given shape can be suitably placed using clonal selection algorithm has discussed. The experimental results and the discussion of them have been given in Sect. 4. Finally, in Sect. 5, we present our conclusions.

2 Clonal Selection Principle and Clonalg Algorithm

Clonal selection is the theory used to explain the basic properties of an adaptive immune response to an antigenic stimulus. It establish the idea that only those cells capable of recognizing an antigenic stimulus will proliferate and differentiate into effector cells, thus being selected against those that do not. The mainly features of the clonal selection principle are affinity proportional reproduction and mutation. The higher affinity, the higher number of offspring generated. The mutation suffered by each immune cell during reproduction is inversely proportional to the affinity of the cell receptor with the antigen. The standard genetic algorithm doesn't account for these immune properties [9].

De Castro & Von Zuben proposed a Clonal selection algorithm named Clonalg, to fulfill these basic processes involved in clonal selection principle. It will be initially proposed to perform machine-learning and pattern recognition tasks, and then adapted to be applied to optimization problems [8]. The algorithm of it for the optimization task is given below.

1. Generate j antibodies randomly.
2. Repeat a predetermined number of times:
 - a. Determine the affinity of each antibody (Ab). This affinity corresponds to the evaluation of the objective function.
 - b. Select the n highest affinity antibodies.
 - c. The n selected antibodies will be cloned proportionally to their affinities, generating a repertory C of clones: the higher affinity the higher number of clones and vice versa.
 - d. The clones from C are subject to hypermutation process inversely proportional to their antigenic affinity. The higher affinity, the smaller mutation, and vice versa.
 - e. Determine the affinity of the mutated clones C .

- f. From this set C of clones and antibodies, select the j highest affinity clones to compose the new antibodies' population.
 - g. Replace the d lowest affinity antibodies by new individuals generated at random.
3. End repeat [7].

3 Computation of the Membership Functions Using Clonalg

The most appropriate placement of membership functions with respect to fuzzy variables can be found for a fuzzy system whose rules table and shape of membership functions were given previously. There are no restrictions for shape of membership functions. They can be generally used but it can be mathematical function whose model is known. Then, the issues to be determined are the parameters that define the model. Because of this, the membership optimization problem can be reduced to parameter optimization problem. How the membership functions compute as a parameter optimization problem using Clonalg is described below for single input-output system .

It is assumed that there are two different membership functions for input and output of considered system and their shapes are right triangle. The membership functions are called $input(x)$ and $output(y)$; x uses slow and fast; y uses easy and difficult. In this case, the linguistic rules are as follows:

Rule 1: *If x is slow then y is easy*

Rule 2: *If x is fast then y is difficult*

If the range of input variable x as [0-7] and the range of output variable y as [0-49] are assumed, then the membership functions of fuzzy system for input and output variables will be as shown in Fig. 1.

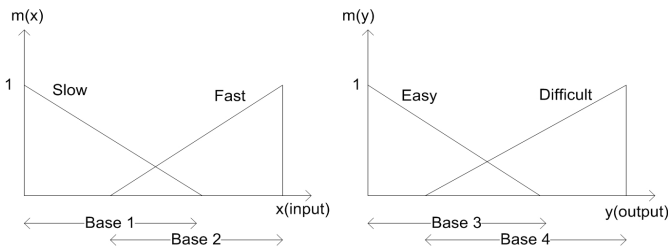


Fig. 1. The membership functions of a fuzzy system for input and output

Input : $x_i = \{1, 3, 5, 7\}$ **Output :** $y_i = \{1, 9, 25, 49\}$, $i=1, 2, 3, 4$

Expected from a Clonalg is to find the base lengths of right triangles. If the base length of each membership function is represented by 6-bit, the genes $Base1Base2Base3Base4$ containing solution of problem has $6*4=24$ bits. In this case the maximum value each base can take is $2^6-1=63$. The domain intervals for input and output variables are [0-7] and [0-49], respectively. The base values are reflected under these values. This is formulated in (1).

$$X_i = X_{\min} + \frac{d}{(2^L - 1)} * (X_{\max} - X_{\min}) \tag{1}$$

L is the length of related variable (in this case, it is Antibody- Ab) in bits, d is the decimal value of this variable, X_{\min} is the minimum value of region to be transformed, and X_{\max} is the maximum value of this region, then X_i is the transformed figure of that variable.

Thereafter, the necessary thing to be done is to create the initial antibody population- Ab with 10 antibodies randomly, and let begin Clonalg process. Table 1 expresses the initial antibody population. Columns 2-5, are the decimal values of lengths of related antibodies. Columns 6-9 are the cases of these lengths reflected to domain intervals. Before the computation of affinity, total error is calculated in (2).

$$\text{Total_Error} = \sum_{i=1}^4 (y_i - y_{\text{clonalg}_i})^2 \tag{2}$$

Where y_i is the output of i^{th} reference input; y_{clonalg_i} which is obtained by Clonalg, is output for i^{th} reference input. The aim is to approximate the total error to zero as close as possible. Affinity function is converted to $4480 - \text{Total_Error}$, in this way; minimization process is also converted to maximization process (Column 10). In order to prevent affinity function from getting negative values, the maximum number 4480 is used and this number is also maximum error at the same time. It is evaluated by $\sum_{i=1}^4 (y_i - 49)^2$; where 49 is the biggest output value for :

Base1<First Ref. Input($x=1$) and $(7 - \text{Base2})$ <First Ref. Input($x=1$) and Base4=0.

Table 1. Initial Ab Population

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|--|-----|-----|-----|-----|--------|--------|--------|--------|----------|---------|
| Antibodies | B1 | B2 | B3 | B4 | B1 ref | B2 ref | B3 ref | B4 ref | Affinity | Cl. Nm. |
| 010100110010111010001000 (Ab_1) | 20 | 50 | 58 | 8 | 2,2 | 5,5 | 45,1 | 6,2 | 2372,3 | 0 |
| 010111100001101001101000 (Ab_2) | 23 | 33 | 41 | 40 | 2,5 | 3,6 | 31,8 | 31,1 | 4217,8 | 1 |
| 111110110001100100000100 (Ab_3) | 62 | 49 | 36 | 4 | 6,8 | 5,4 | 28 | 3,1 | 4438,5 | 3 |
| 101001101001011011111010 (Ab_4) | 41 | 41 | 27 | 58 | 4,5 | 4,5 | 21 | 45,1 | 4449,2 | 5 |
| 100010111010110110011001 (Ab_5) | 34 | 58 | 54 | 25 | 3,7 | 6,4 | 42 | 19,4 | 3461,8 | 0 |
| 01011110100011011111011 (Ab_6) | 23 | 52 | 27 | 59 | 2,5 | 5,7 | 21 | 45,8 | 4294,3 | 2 |
| 111110110010100010111110 (Ab_7) | 63 | 25 | 17 | 30 | 7 | 2,7 | 13,2 | 23,3 | 2946,1 | 0 |
| 011110010010010101001010 (Ab_8) | 30 | 18 | 21 | 10 | 3,3 | 2 | 16,3 | 7,78 | 4169,1 | 1 |
| 100011110100110110111110 (Ab_9) | 35 | 58 | 27 | 30 | 3,8 | 6,4 | 21 | 23,3 | 4127,9 | 1 |
| 010110011011110111010001 (Ab_{10}) | 22 | 27 | 55 | 17 | 2,4 | 3 | 42,7 | 13,2 | 3896,2 | 0 |

The solution of first antibody is seen in Fig. 2 for initial population. The base values for x are 2.22 and 5.56. In the same way, the base values for y are 45.11 and 6.22. In this case, output is evaluated for any input reference such that, grade of membership of input is 0.549 for assumed input reference at $x = 1$. At given fuzzy system rules, “If x is slow then y is easy” is seen. If y value is directly assumed as defuzzification processes, according to this case, y point of easy membership function, whose

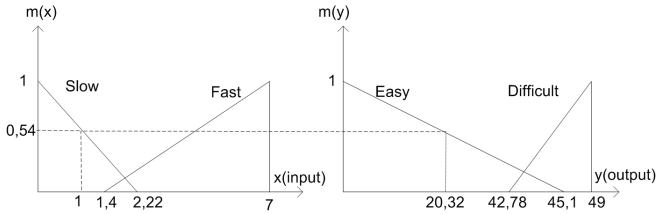


Fig. 2. The solution of the first chromosome in the initial population

degree is 0.549 is found, that is, it is 20.32 whereas the real output value is 1. So, error at this point is $(1-20.32)^2 = 373.26$. Other error values are calculated and all of them are accumulated. Then this accumulation is subtracted from 4480 for acquired the affinity value.

After acquiring the all affinity values, they are sorted descending order. Assume that the parameter n is 8. Then the n highest affinity elements are selected and number of clones of Ab_i is calculated proportionally to their affinity (Column 11) using the given formula: $N_c = \sum_{i=1}^n \text{round}(\beta.N/i)$. This implements the affinity proportional reproduction; the higher the affinity, the higher the number of clones. While Ab_4 in the fourth row which has the maximum affinity value in the population is cloned five times, no clone is generated of the Ab_5 in the fifth row.

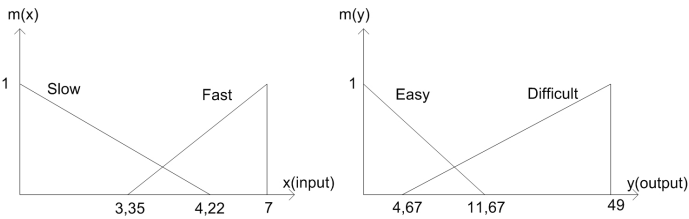


Fig. 3. The membership function shape of the optimal solution

After cloning process, new Ab population is mutated. The inverse of an exponential function can be used to establish the hypermutation rate $\alpha(.)$ described in [9]. For example each of the Ab_4 and its clones suffer a two bit mutation. But Ab_5 and its clone suffer a six bit mutation. After 20 iteration completed, the individual whose fitness or affinity is maximum has been accepted as optimal solution.

The optimal solution of this optimization problem obtained from Clonalg is given in the Table 2 and its membership function shape is depicted in Fig. 3.

Table 2. The Optimal Solution

| Antibodies | B1 ref | B2 ref | B3 ref | B.4 ref | y1 (x=1) | y2 (x=3) | y3 (x=5) | y4 (x=7) | Affinity |
|--------------------------|-----------|-----------|-----------|------------|-------------|-------------|-------------|-------------|----------|
| 100110100001001111111001 | 4,22 | 3,65 | 11,67 | 44,33 | 2,763 | 8,289 | 24,82 | 49 | 4476,4 |

4 Experimental Results

In this section, Clonalg and GA used to optimize the fuzzy membership functions were implemented by Matlab 7.1 R14 and compared empirically. Results showed that CLOANLG was more effective method than GA for determining the membership functions. For showing the results weren't obtained by coincidentally, 20 different groups were generated. A group consisted of 10 different initial populations and a population consisted of 10 *Ab* individuals. The algorithms were set to run for 20 generations on each population in the groups, respectively. Then the results are compared group by group.

Because of two different algorithms performed on the same populations, a paired data test has been used to compare them statistically. If distributions of population individuals were normal, the parametric paired t-test could be used. But we didn't have enough knowledge about distributions of population. So, the *Wilcoxon Signed-Rank Test for Paired Data* which is a nonparametric alternative to the paired t-test was used to compare them. It examined the truth of the H_0 hypothesis according to its *p* value for a definite significance level. For example, if *p* test value is bigger than 0.05, then H_0 is accepted for the significance level %95. For this work H_0 was constructed as follow: "There is no statistical significant difference between Clonalg and GA for the significance level %95" The test performed on each groups and the *p* values acquired from tests were showed in Table 3. As shown, H_0 hypothesis was accepted for only one group in the tenth column. For others, it was rejected. So we could say clearly, there was a statistical significant difference between them for the significance level %95 for this work.

Table 3. *p* values of Wilcoxon Signed-Rank Test for Paired Data according to groups

| Gr | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|----------|------|------|-----|-----|------|-----|------|------|-----|-----|------|------|-----|------|-----|-----|------|-----|------|-----|
| <i>p</i> | .009 | .001 | .02 | .02 | .001 | .02 | .003 | .005 | .02 | .08 | .001 | .001 | .04 | .003 | .01 | .02 | .001 | .02 | .001 | .04 |

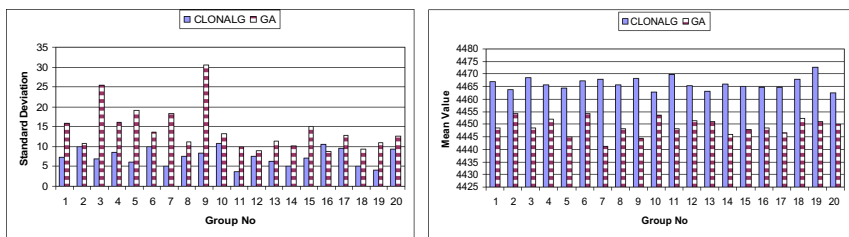


Fig. 4. Some descriptive statistics of groups: Stand. Deviation (*left plot*) and Mean (*right plot*)

Via the Wilcoxon Test, it was showed that the algorithms were different but still it wasn't known which one was better than the other. For understanding that, some descriptive statistics –standard deviation and mean value- was used. In the left sides of Fig. 4, standard deviations of the groups were depicted. As seen that Clonalg's standard deviation of groups were lower than the GA's for all groups. In addition that its

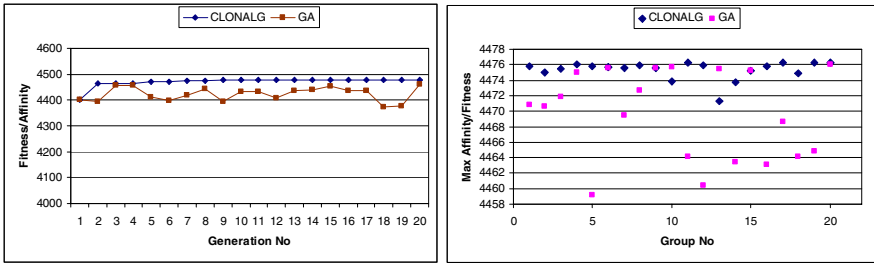


Fig. 5. Fitness/Affinity functions value of a population (*left plot*) and Max affinity/fitness values of the each group (*right plot*)

mean values of groups are higher than the GA’s for all groups too. The higher mean value was a satisfactory case, because it was a maximization problem. According to these knowledge, it could be said that using CLOANLG was advantageous than using GA for finding optimum values of fuzzy membership functions.

These algorithms were also compared from a different aspect without using a statistical test. Left plot of the Fig. 5 showed that fitness/affinity values of algorithms according to generation number of a population. As seen from that plot, Clonalg converged to optimum solution faster than GA and its affinity values were more stable, also. In the right side of the Fig. 5, max affinity or fitness values of the each group were depicted. According to them, it was clearly seen that Clonalg was more successful than GA to find the optimum solutions. Because GA converged to the max value of this problem in six groups from these 20 groups and only in two of them, it succeeded to generate better solutions than the Clonalg’s solution, in the others it found the same values as Clonalg’s. As for the 14 groups rest, Clonalg found the best solutions. Than it was straightforward to say the Clonalg represented a good performance while finding the optimum base distance of a fuzzy membership functions.

5 Conclusion

The most appropriate placement of membership functions with respect to fuzzy variables can be found for a fuzzy system whose rules table and shape of membership functions were given previously. There are no restrictions for shape of membership functions. They can be generally used but it can be mathematical function whose model is known. Then, the issues to be determined are the parameters that define the model. Because of this, the membership optimization problem can be reduced to parameter optimization problem. How the membership functions compute as a parameter optimization problem using Clonalg was described in this work.

Clonalg used to optimize the fuzzy membership functions was implemented by Matlab 7.1 R14. Then GA was coded too in the same conditions for comparing with clonal selection algorithm. Firstly, 20 different groups were generated for showing the results weren’t obtained by coincidentally. A group consisted of 10 different initial populations and a population consisted of 10 *Ab* individuals. The algorithms were set to run for 20 generations on each population in the groups, respectively. Then the acquired results are compared group by group.

For 20 groups, the Wilcoxon Signed-Rank Test examined a H_0 hypothesis constructed as follows: “ H_0 : There is no statistical significant difference between Clonalg and GA for the significance level %95”. This hypothesis was rejected for 19 groups. So, we could say clearly, there was a statistical significant difference between Clonalg and GA for significance level %95 for this work. For understanding which one was better than the other, some descriptive statistics –standard deviation and mean value– was used. It was found that Clonalg’s standard deviations of groups were lower than the GA’s and its mean values of groups were higher than the GA’s for all groups. Also, it converged to optimum solution faster than GA and found the optimum solutions more successfully.

As a conclusion, it is shown that Clonalg can be used while determining optimum values of membership functions for a fuzzy system whose rule table and model of membership function’s shape are known and using of it is advantageous than using one of the best global search algorithm GA.

Acknowledgements. The authors acknowledge the support of this study provided by Selçuk University Scientific Research Project Fund for project no: 07701049.

References

1. Arslan, A., Kaya, M.: Determination of fuzzy logic membership functions using genetic algorithms. *Fuzzy Sets and Systems*, 118 (2001), 297–306.
2. Cheng, H.D, Lui, Y.M.: Automatic Bandwidth Selection of Fuzzy Membership Functions. *Information Sciences*, 103 (1997), 1-27.
3. Bağış, A.: Determining fuzzy membership functions with tabu search – an application to control. *Fuzzy Sets and Systems*, 139 (2003), 209-225.
4. Cerrada, M., Aguilar, J., Colina, E., Titli, A.: Dynamical membership functions: an approach for adaptive fuzzy modeling, *Fuzzy Sets and Systems*, 152 (2005), 513–533.
5. Simon, D.: H_∞ estimation for fuzzy membership function optimization. *International Journal of Approximate Reasoning*, 40 (2005), 224–242.
6. Yang, C-C., Bose, N.K.: Generating fuzzy membership function with self-organizing feature map. *Pattern Recognition Letters*, 27 (2006), 356-365.
7. Cortes, N. C., Perez, D.T., Coello, C.A.: Handling Constraints in Global Optimization Using an Artificial Immune System. *Lecture Notes in Computer Science Vol. 3627*, Springer-Verlag, Berlin Heidelberg, (2005), 234-247.
8. De Castro, L.N., Zuben, J.V.: Learning and Optimization Using Clonal Selection Principle. *IEEE Transaction on Evolutionary Computation*, Special Issue on Artificial Immune Systems, 6,3 (2002), 239-251
9. De Castro, L.N., Timmis, J.I.: *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer -Verlag, London, (2002), 357 pages.
10. Karr, C.L.: Design of an Adaptive Fuzzy Controller Using a Genetic Algorithm. *Proc. of the 4th Intl. Conf. on Genetic Algorithms*, (1991).
11. Lee, M.A., Takagi, H.: Integrating design stages of fuzzy systems using genetic algorithms. *2nd IEEE Intl. Conf. On Fuzzy Systems*, (1993).
12. Meredith, D.L., Karr, C.L., Kumar, K.: The use of genetic algorithms in the design of fuzzy logic controllers. *3rd Workshop on Neural Network WNN’92*, (1992).

Clustering of Leaf-Labelled Trees^{*}

Jakub Koperwas and Krzysztof Walczak

Institute of Computer Science, Warsaw University of Technology,
Nowowiejska 15/19, 00-665 Warsaw, Poland

J.Koperwas@elka.pw.edu.pl, K.Walczak@ii.pw.edu.pl

Abstract. This paper introduces novel methodology for the clustering of data represented as leaf-labelled trees on the same leaf-set. We define an abstract term - the representative tree, which can be represented with a variety of trees, depending on applications. The quality of tree-clustering is based on Information Gain, which measures the increase of information contained by representative trees of the resulting clusters compared to a single representative tree of the whole dataset. Finally, we propose the k -best algorithm the objective function of which is to maximize the information gain. We show how it can be constructed for two different representative trees, well-known in phylogenetic analysis. Developed algorithms yield very promising results.

1 Introduction

Existing data mining techniques concerning clustering concentrate on tabular (attribute-value) data. However, many data or processes from various fields of interests i.e. bioinformatics, telecommunications, text mining or image processing are represented as trees. Thus there is a need to adopt data-mining techniques (including clustering algorithms) to be applicable to tree data in such a way that they provide reliable results. Some aspects of tree mining were already discussed in literature, among others: mining frequent patterns in trees [11] and optimal partitioning of a tree, called "clustering on trees" [3].

This paper is part of a work in progress devoted to the clustering of data represented as trees. The single work devoted to the problem of clustering of data represented as trees was presented by [10]. The authors concentrated on phylogenetic trees, which are a special case of leaf-labelled trees, because they are binary. The authors discussed various clustering algorithms and measure their quality for the clustering of phylogenetic trees. The quality measure provided by authors is based on statistical criteria and is only applicable for phylogenetic trees.

We develop the general approach for trees which are not necessarily binary trees, and can also be extended to trees with a different set of leaves, which increases the potential applicability of our technique. However, in this paper we

^{*} The research has been partially supported by grant No 3 T11C 002 29 received from Polish Ministry of Education and Science.

focus only on those with the same leaf set. The leaf-labelled trees may be used where the partition of problem space needs to be represented.

Our quality measure is based on structural properties of trees rather than on a distribution probability criterion specific for particular problem. Our approach stays at the same time as general as possible and allows the modifications towards any specific dataset. It is achieved thanks to an abstract representative tree term, which can be selected freely depending on a problem being analyzed. The quality of tree clustering is based on Information Gain. It measures the increase of information contained by the representative trees of resulting clusters with respect to a single representative tree of the whole dataset. Finally we propose the k-best algorithm the objective function of which is to maximize the information gain. We show how it can be constructed for two different representative trees which are well-known in phylogenetic analysis.

2 Basic Notions

Leaf-labelled trees are trees with labels attached to their leaves. Trees may be either rooted, if there is one distinguished node called root, or unrooted which is the more general case. If a tree has assigned an order to its nodes it is called an ordered tree, otherwise it is unordered, which is the more general case. Unrooted trees may be represented by a set of splits with respect to each edge. Rooted trees are often represented with clusters of leaves.

Split/Bipartition of a tree T with leafset L with respect to edge an edge e is a partition of leafset L into two sets A and B that occur when the edge e is removed. $A|B$ is a bipartition (split) of L corresponding to edge e . If $|A|$ or $|B|$ is equal to 1, the split is trivial. For a tree from Fig. [1](#) a) the splits are: $a|bcdef, b|acdef, c|abdef, d|abcef, e|bcdcf, f|abcde, ab|cdef, abc|def, abcd|ef$

Due to lack of space we do not discuss a variety of distance measures. We describe here Robinson - Fould's Distance as best-suited for leaf-labelled trees with the same leaf-set.

Robinson - Fould's distance originated from phylogenetic analysis. It is defined as the proportion between the number of splits not shared by compared trees. As it was proposed for phylogenetic trees, it is defined for leaf-labelled trees with the same leaf-set. R-F distance between two trees T_1 and T_2 with set of splits S_1 and S_2 respectively is as follows:

$$d_{R-F}(T_1, T_2) = |S_1 \cup S_2| - |S_1 \cap S_2|. \quad (1)$$

For the trees presented on Fig. [1](#) $d_{R-F}(T_1, T_2) = 2$.

The R-F distance can be normalized with number of leaves, however for leaf-labelled trees on the same leafset it is not necessary, therefore it will be omitted due to the clearance of presentation.

The strict consensus tree is defined in terms of splits. Strict consensus tree is a tree constructed of all splits common to all trees in a given profile of trees. Fig. [1](#) presents two trees together with their strict consensus tree. The

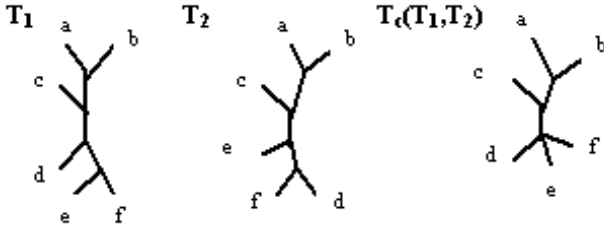


Fig. 1. Two leaf-labelled trees together with their strict consensus tree

common splits of those trees, that build the strict consensus tree are as follows: $a|bcdef, b|acdef, c|abdef, d|abcef, e|abcdef, f|abcde, ab|cdef, abc|def$.

Because the concept of a consensus tree is very strict, for many trees, a consensus tree can easily become a star (a tree built only of trivial splits). In order to deal with this problem, many variations of consensus trees were proposed, among others, a majority rule consensus tree. **Majority rule consensus tree** is built of splits that occur in the majority of trees.

3 Clustering Quality Assessment

Quality measures, designed for tabular data, cannot be used for assessing tree-clustering. Therefore we define our measure on the basis of structural information contained in trees. Our measure is based on the belief that it is required that the examples belonging to the same cluster share common, retrievable knowledge. This way it provides a good and useful clustering. The basis of our approach is the representative tree and its informativity.

Definition 1. *Representative tree - a tree that shares common knowledge of all trees in a cluster. The representative tree is built on a superset of splits of all trees in a group. The representative tree may not introduce any new information, besides that contained by input trees:*

$$S_R = f(S_1, \dots, S_n) \quad S_R \subseteq \bigcup_{i \in C} S_i, \tag{2}$$

where S_i is a set of splits of tree T_i and S_R is a set of splits of a representative tree.

Note that this term is as general as possible and does not imply a method of construction of such a tree. We decided to provide such a term in order to stay independent of any particular application of a clustering approach. In this paper we discuss a strict consensus tree and a majority rule consensus tree as the possible representative trees, however, we neither select the best of them nor limit the possible choice to them. In order to assess the quality of clustering we, as the simplest approach, measure the amount of common information in each cluster, i.e. the amount of information located in the representative trees.

In order to measure the amount of information (the informativity of trees), we use a measure presented in [6].

Definition 2. *Informativity of the tree T is the number of non-trivial splits contained by this tree:*

$$I(T) = |S^{nt}|, \tag{3}$$

where $|S^{nt}|$ is the amount of non-trivial splits

Because in leaf-labelled trees on the same leafset, the number of trivial splits in each tree is constant, we will omit it in all of the equations for the clarity of presentation. In order to illustrate the term of *informativity of a tree*, we have to recall a strict consensus tree construction procedure. If the split that was not present in all the trees it was not inserted into a strict consensus tree. This way the information contained in the non-trivial split was lost. In the pessimistic case of there being no information between trees, the strict consensus tree will be a perfect star $I(T_C) = 0$. Basing on the informativity of a tree we define the quality measure.

Definition 3. *Cluster Information Loss. The amount of information, that is lost when we replace the cluster of trees, with one representative tree:*

$$\Delta I_{C_x} = \sum_{i=1}^l |S_i \div S_R|. \tag{4}$$

The information loss defined has following properties:

1. It has a value of 0, if all trees in a cluster are identical and replaced by the same tree. Additionally, it reflects the frequency of removed splits.
2. It does not judge the input data. If the input trees contain no informative trees, but are identical, the information loss will still be 0.
3. If any particular input tree has fewer splits than representative tree, it is also considered a loss.

If we choose strict consensus tree as representative tree it can be shown that the information loss can be formulated as follows:

$$\Delta I_{C_x} = \left(\sum_{i=1}^l I(T_i) \right) - l * I(T_C), \tag{5}$$

where l is the number of trees assigned to a cluster.

Definition 4. *Clustering Information Loss - sum of information losses in all clusters*

$$\Delta I = \sum_{i=1}^k \Delta I_{C_i} \tag{6}$$

For the comparison of the efficiency of algorithms on a single dataset, the information loss is sufficient measure, but if multiple datasets are involved, we have to measure the quality with respect to the input data. Therefore we define information gain.

Definition 5. *Information Gain*

$$IG = \frac{\Delta I_{C_0} - \Delta I}{\Delta I_{C_0}}, \tag{7}$$

where ΔI_{C_0} is the information loss of replacing whole profile of trees with one representative tree.

4 K-Best Tree Clustering Algorithm

Here we define the k-best problem and provide algorithms for two most intuitive representative trees: a strict consensus tree and a majority rule consensus tree.

Definition 6. *K-best problem is the problem of finding the partition of dataset on k clusters, (where k is a given value), in such a way that this partition maximizes Information Gain towards a given type of a representative tree.*

4.1 K-Best for Majority Rule Consensus Tree as Representative Tree

Here we show that k-mean algorithm maximizes Information Gain with respect to the majority rule consensus tree. In order to apply k-mean algorithm to tree data we must provide a distance measure between two trees and a centroid tree for a group of trees. Centroid by definition is the point that minimizes average dissimilarity to all points in a cluster. [4] shows that the majority rule consensus tree is in fact a median tree (which has the same definition as centroid but defined for trees). Therefore the majority rule consensus tree has to be taken as centroid in order to use k-mean algorithm. The authors in [10] used the strict consensus tree as the Centroid. However, such an approach cannot be justified as this tree does not meet the conditions for centroid. The objective function of k-mean algorithm is defined as follows:

$$\min_{C, \{y_k\}_{k=1}^K} \sum_{k=1}^K \sum_{C(i)=k} d(x_i, y_k) = \min_{C, \{T_{M_k}\}_{k=1}^K} \sum_{k=1}^K \sum_{C(i)=k} d(T_i, T_{M_k}), \tag{8}$$

where T_M is majority rule consensus for a given cluster, and $C(i)$ is a function that assigns tree i to a cluster:

$$\Delta I(T_M) = \sum_{k=1}^K \sum_{C(i)=k} |S_i \div S_{M_k}|, \tag{9}$$

and because $S_i \cap S_{M_k} \subseteq S_i \cup S_{M_k}$,

$$\Delta I(T_M) = \sum_{k=1}^K \sum_{C(i)=k} d(T_i, T_{M_k}). \tag{10}$$

So this modified k-mean minimizes the information loss and because ΔI_{C_0} is the constant value for a given input data this algorithm also maximizes Information Gain, i.e. quality measure. Therefore the objective function of k-mean with majority rule consensus tree as a centroid tree is to solve k-best problem with respect to the majority rule consensus tree as a representative tree.

4.2 K-Best for Strict Consensus Tree as Representative Tree

For k-mean algorithm, it can be shown that replacing centroid tree with strict consensus causes, that in general, algorithm does not converge to local optimum. Therefore we present a solution for solving k-best problem with respect to strict consensus tree, with the usage of agglomerative clustering. We make two significant changes in the agglomerative clustering algorithm:

1. Replace the most popular merging strategies: minimum, maximum or complete linkage with our *minimum-loss linkage*, which is realized with modification of R-F distance measure.
2. We represent a cluster with only strict consensus tree and a number of trees assigned. We can do this because the information loss requires only a strict consensus tree to be counted. We can obtain consensus tree of a cluster that originated from merging due to the following property:

$$T_C(T_C(T_1, T_{n-1}), T_n) = T_C(T_1, T_n), \tag{11}$$

where T_C is a strict consensus tree.

Our goal is to minimize ΔI , and because this value is counted on the basis of strict consensus tree here, we represent clusters as strict consensus trees. In each iteration of algorithm, we merge two clusters together. Each merging causes that information loss of a given clustering is getting worse, which is normal. Our goal is to choose clusters to merge in each iteration in such a way it minimizes the information lost after merging. Counting information loss for all possible mergings in each iteration is highly inefficient. Therefore we just count the delta because it allows to reduce the problem to the comparing of only two clusters at a time, as the sum of information in uninvolved clusters remains the same. Therefore the merging condition is:

$$\arg \min_{C_i, C_j} \Delta I' - \Delta I, \tag{12}$$

$$\Delta I' - \Delta I = (\sum_{C(i)=z} I(T_i) - l_z * I(T_{C_z})) - ((\sum_{C(i)=x} I(T_i) - l_x * I(T_{C_x})) + (\sum_{C(i)=y} I(T_i) - l_y * I(T_{C_y}))), \tag{13}$$

Because the cluster z is the one that emerged from clusters x, y and because of the property of strict consensus tree: (2)

$$\begin{aligned} \Delta I' - \Delta I = & l_x * I(T_{C_x}) + l_y * I(T_{C_y}) - l_z * I(T_{C_z}) = \\ & l_x * |S_{C_x}| + l_y * |S_{C_y}| - l_z * |S_{C_z}| = \\ & l_x * |S_{C_x}| + l_y * |S_{C_y}| - (l_x + l_z) * (|S_{C_x} \cap S_{C_y}|) = \\ & l_x * (|S_{C_x}| - |S_{C_x} \cap S_{C_y}|) + l_y * (|S_{C_y}| - |S_{C_x} \cap S_{C_y}|). \end{aligned} \tag{14}$$

So in order to make the best merging decision in each step we need to minimize the expression (14), where each cluster is represented by only one tree. In subsequent iterations we need only to keep the actual strict consensus trees for each cluster and the number of trees assigned to it. Algorithm stops when it reaches the given k number of clusters.

5 Results

We have performed clustering for two datasets of phylogenetic trees kindly provided by Li-San Wang. The first dataset pevccal contains 168 trees, the other camp contains 216 trees. Both datasets contain trees that are very similar. In first experiment we have used the majority rule consensus tree as a representative tree and compared the results of k-mean, the objective function of which is to maximize information gain with hierarchical clustering with three well-known strategies: minimum(Agg-min), maximum(Agg-max) and complete(Agg-compl) linkage. In the other experiment we have used the strict consensus tree as a representative tree. We have compared our minimum loss strategy(Agg-min) of hierarchical clustering with min, max and complete-linkage, additionally we have compared it with k-mean approach, where we replace the middle tree with the strict consensus tree(k-str) as in (10). Both of the discussed algorithms perform extremely well on both datasets. On the larger dataset(camp) the minimum loss strategy occurred to be at minimum 11% and up to 92% better than other algorithms. The k-mean algorithm occurred to be at minimum 44% and up to

Table 1. Results of clustering with proposed algorithms

| | | majority rule consensus | | | | strict consensus | | | | |
|---------|--------|-------------------------|---------|-----------|---------|------------------|---------|-----------|-------|--|
| K | K-mean | Agg min | Agg max | Agg compl | Agg inf | Agg min | Agg max | Agg compl | K-str | |
| pevccal | | | | | | | | | | |
| 2 | 0.37 | 0.36 | 0.36 | 0.36 | 0.36 | 0.33 | 0.33 | 0.33 | 0.31 | |
| 3 | 0.57 | 0.57 | 0.57 | 0.57 | 0.63 | 0.63 | 0.63 | 0.63 | 0.63 | |
| 4 | 0.61 | 0.65 | 0.65 | 0.65 | 0.67 | 0.67 | 0.67 | 0.67 | 0.66 | |
| 5 | 0.68 | 0.66 | 0.66 | 0.66 | 0.71 | 0.70 | 0.70 | 0.70 | 0.69 | |
| 6 | 0.69 | 0.67 | 0.68 | 0.67 | 0.75 | 0.72 | 0.72 | 0.73 | 0.70 | |
| 7 | 0.71 | 0.68 | 0.71 | 0.69 | 0.77 | 0.72 | 0.75 | 0.74 | 0.73 | |
| 8 | 0.74 | 0.69 | 0.72 | 0.69 | 0.78 | 0.75 | 0.77 | 0.75 | 0.73 | |
| camp | | | | | | | | | | |
| 2 | 0.08 | 0.00 | 0.02 | 0.00 | 0.03 | 0.00 | 0.01 | 0.00 | 0.00 | |
| 3 | 0.25 | 0.00 | 0.04 | 0.01 | 0.06 | 0.00 | 0.03 | 0.00 | 0.00 | |
| 4 | 0.25 | 0.01 | 0.06 | 0.04 | 0.09 | 0.01 | 0.06 | 0.00 | 0.04 | |
| 5 | 0.27 | 0.01 | 0.09 | 0.08 | 0.11 | 0.01 | 0.07 | 0.00 | 0.08 | |
| 6 | 0.27 | 0.02 | 0.12 | 0.12 | 0.14 | 0.01 | 0.11 | 0.00 | 0.04 | |
| 7 | 0.29 | 0.02 | 0.14 | 0.15 | 0.16 | 0.02 | 0.12 | 0.02 | 0.10 | |
| 8 | 0.32 | 0.03 | 0.16 | 0.18 | 0.18 | 0.03 | 0.14 | 0.07 | 0.08 | |

99% better than the other algorithms. On the smaller dataset(pevcca1) all the compared algorithms have similar results, however in most cases our algorithms were slightly better. Only in one case (for k-mean) our algorithm was at most 6% worse than the others. It occurred that for improper k-mean, (where a strict consensus is used), algorithm may sometimes never stop, but for the proper usage (with majority consensus) it behaves perfectly, which confirms our theory. Neither of the discussed algorithms finds the global optimum, and no other algorithm, but they provide excellent results.

6 Discussion

In this paper we have defined the quality measure and k-best problem. We have provided two algorithms that solve that problem for different representative trees and achieved excellent results. We have focused on leaf-labelled trees on the same leaf-set, which is a part of a larger approach of clustering tree data. The unrooted, unordered trees were used in our reasoning as it is the most general case, however, rooting and ordering can be easily included. In future, the approaches of clustering leaf-labelled trees on any leaf-set, and node-labelled trees will be presented. The works on generalized k-best algorithm are also in progress. Moreover, those methods need to be verified on a larger set of real-life problems.

Acknowledgments

We kindly thank Li-San Wang for providing us with datasets.

References

1. Akutsu, T., and Halldrsson, M. "On the approximation of largest common point sets and largest common subtrees", Unpublished manuscript (1997).
2. Amenta, N., Clarke, F., and St. John K. "A Linear-time Majority Tree Algorithm" WABI (2003) 216-227
3. Auber, D. and Delest, M. "A clustering algorithm for huge trees" Advances in Applied Mathematics 31 (2003) 46-60
4. Barthelemy, J.P., McMorris, F.R. "The median procedure for n-trees". J. Classif. 3 (1986) 329-334
5. Bille, P. "Tree Edit Distance, Alignment Distance and Inclusion", Technical report TR-2003-23 in IT University Technical Report Series (2003)
6. Bryant, D. "Building Trees, Hunting For Trees, And Comparing Trees. Theory And Methods In Phylogenetic Analysis." Ph.D Thesis University of Canterbury (1997)
7. Elias, I. "Settling the Intractability of Multiple Alignment" Proc. of the 14th Ann. Int. Symp. on Algorithms and Computation ISAAC (2003) 352-363
8. Jansson, J. and Lingas, A. "A Fast Algorithm for Optimal Alignment between Similar Ordered Trees". Lecture Notes in Computer Science 2089 (2001) 232-232

9. Amenta, N. and Klingner, J. "Case study: Visualizing sets of evolutionary trees." 8th IEEE Symposium on Information Visualization (2002) 71-74
10. Stockham, C., Wang L.S. and Warnow T. "Statistically Based Postprocessing of Phylogenetic Analysis by Clustering". *Bionformatics* 18 (2002) 285-293
11. Xia, Y. et. al. "Mining Closed and Maximal Frequent Subtrees from Databases of Labeled Rooted Trees" *IEEE Transactions on Knowledge and Data Engineering* 17 (2005) 190-202

Social Organization of Evolving Multiple Classifier System Functioning in Changing Environments

Sarunas Raudys

Vilnius Gediminas Technical University
Sauletekio 11, Vilnius, LT-10223, Lithuania
raudys@ktl.mii.lt

Abstract. We model populations of classifiers which are aimed to function in permanently varying environments, adapt to unexpected changes, to comply fitness function and survive. A failure to fulfill survivability condition is resulting in unsuccessful agents being removed from the agent society and be replaced by newborns which inherit some upbringing learning information from parent agents. We split the agent population into groups and suggest storing agent's gains accumulated during most recent periods, distort randomly training signals and a level of survival threshold. A presence of optimal number of groups and a necessity of small groups with mutually collaborating agents is demonstrated.

1 Introduction

The necessity to adapt rapidly to changes of environments becomes one of most important obligations while developing modern robots and intelligent computer programs [1]. In analysis of social, trade and industry problems we face classification tasks and the need of fast adaptability too. Different combinations between artificial neural networks (ANN's) and evolutionary algorithms, including using latter algorithms to evolve ANN connection weights, architectures, learning rules, and input features often are used simultaneously [2]. The combination of evolution and learning may induce better results, desirably reaching global optima [2], [3], [4], [5].

Assortment, or the degree of segregation of different types of individuals into different groups, plays a central role in social evolution [6]. Different mathematical frameworks for studying social evolution use different terminology, but all agree on the central role of assortment [6], [7], [8], [9] [10], [11], [12]. In major part of simulation studies, the intelligent agents are not adaptive. *Single, static* pattern recognition (PR) task is used to train decision making algorithms regularly. One of few exceptions is a series of conference papers [13], [14], [15], [16] where sequences of large number of different PR tasks with diverse characteristics were used to train *the populations of the classifiers*. In order to understand *basic regularities of adaptation in changing environments*, at first the simplest training methods have to be investigated in situations where PR tasks are alternating for a long time.

In present paper we model intelligent adaptive agents as nonlinear single layer perceptrons (SLPs) trained by total gradient algorithm. Following general assertion of chaos theory claiming that many phenomena in micro and macro worlds follow the

same fundamental relationships, we believe that such simple model like SLP would provide opportunity to formulate various general statements, since it has many traits of universality [17]. In [13] two different PR tasks model was suggested to analyze aging problems of intelligent adaptive agents, individuals, groups of individuals or even social groups. In [14] it was shown that a difference between desired outputs (target values) of the perceptron (called “a stimulation”) affects training speed: with an increase in stimulation strength, training speed increases at first, saturates and then starts decreasing. In [15] a feedback chain used to control magnitude of the stimulation was interpreted as “synthetic emotions” which if correctly determined speeds up adaptation process. In [16] long-lasting sequences of numerous alternating pattern recognition tasks were considered. *Each time, training process started from previous weight vector.* It was shown that corrupted training signals assist in faster adaptation of the agents to the PR tasks changes. An artificial population paradigm, where different agents possess diverse levels of a noise injected to training signals was introduced. To help artificial populations of the classifiers to withstand lengthy series of sudden environmental changes, we are investigating the agent populations with offspring and inheritance. This study confirmed that optimal values of the noise level are following alterations of strengths of the PR task changes.

An objective of present paper is the analysis of training speed of the population of the classifiers in situations where pattern recognition tasks are changing permanently and the agents are organized into the groups where they help each other, and do not allow other groups to pass away during “hard times”. Effects of distribution of the agents among the groups and impacts of randomly fluctuating survivability threshold and storing the agents’ earnings accumulated during latest time periods are taken into account. Analysis of or the degree of segregation of different types of individuals into different groups has been performed in a number of previous research papers [6], [7], [8], [9] [10], [11], [12]. *Our novelty is investigation of these problems in the context of permanent pattern recognition task changes.*

2 Different Classification Tasks Model

The SLP as the agent’s model. Environmental changes. We consider standard *nonlinear* SLP with sigmoid activation function, $output = 1/(1+\exp(-sum))$, where weighted sum of inputs, $sum = w^T w x + w_0$ [18], [19]. Note that a slope (incline) of function $output = f(sum)$ is the highest where $sum = 0$. If sum moves toward \pm infinity, the slope diminishes and approaches zero [13], [19]. Environmental changes are mimicked by altering the pattern recognition tasks that the artificial agents have to solve. Simplicity sake, the classification tasks considered are two-class two-dimensional (2D) Gaussian classes, $N(\mu_1, \Sigma)$, $N(\mu_2, \Sigma)$. We consider a sequence of s_{max} pattern recognition tasks, $PRT_1, PRT_2, \dots, PRT_{s_{max}}$. The power of the changes is varying according to a priori definite rule. We are changing only matrix Σ : $\Sigma = T \begin{bmatrix} 1 & 0.98 \\ 0.98 & 1 \end{bmatrix} T$, where

$T = \begin{bmatrix} \rho & 0 \\ 0 & B(t) \end{bmatrix}$ and $B(s) = \beta(s)^{(-1)^s}$, $s = 1, 2, \dots, s_{max}$. Thus, in turn, after each t_{change} batch iterations, the data are rotated counter clockwise and after another t_{change} iterations –

clockwise. To have the agents behavior somewhat different, each single agent is trained with its own data set where each time, a small Gaussian noise is added to components of mean vectors, $\mu_1 = -\mu_2$ and parameter $\rho=1$.

To find weights, w_0, \mathbf{w} , we use a sum of squares cost function with targets $t_1=0, t_2=1$ and gradient descent algorithm in a batch mode. In training, magnitudes of the weights are increasing [13], [19]. Analytical calculations show that if the weights are small, the gradient is large. If the weights become large, the gradient becomes small. So, if the SLP has learned to solve its task properly and the weights are already large, the perceptron is unable to re-learn a new task quickly. *The weights increase and subsequent decelerating of training process are key aspects of our analysis.*

We suppose that the single layer perceptron (the agent) has a limited number of epochs, t_{\max} , to adapt to PR task change in order to solve new PR task correctly: to diminish classification error until P_{goal} , *a priori* defined survivability threshold. If the agent fails to fulfill this requirement, it perishes. A new agent (offspring) takes its place in the population of agents. The two different pattern classification tasks model [13] was established on two distinct pattern recognition tasks, PRT1 and PRT2. Initially, the perceptron is trained with data of PRT1. The weight vector \mathbf{ws} is obtained. At some time instance, the PR task is changed. At this point, the perceptron is trained with data PRT2, starting from weight vector \mathbf{ws} . If the first training was successful, the components of the weight vector, \mathbf{ws} , became large. Large starting weights, \mathbf{ws} , may be deleterious for efficient training with data PRT2. If training time is limited, the perceptron may fail while attempting to learn solving the task PRT2.

Means to increase the survivability. Accumulation of “earnings”. In order to ensure fast training with new, the changed data, *one needs to control the growth of the weights while training the perceptron.* The weight growth may be held back if a noise is injected into the targets, i.e. the perceptron is trained with partially incorrect targets. We may use other regularization techniques for perceptron training, e.g. a weight decay term [18], [19]. We assume that α_i is specific to each agent. In analysis of social systems, however, a noise injected to targets, α_i , a fraction of *incorrect training signals*, possibly, *could be interpreted* as criminality, changes of laws, consequences of differences in value systems, “minor” environmental catastrophes, economy crises. A noise level, α_i , is affecting the agent’s re-training speed and its ability to survive sudden PR task changes. At the start of each experiment, for all m agents in the population we assigned different values of parameter α_i . Artificial training set (comprising 50 training vectors from each class) is supplied to each agent. Diverse values of α_i lead to various “learning styles” of different adaptive agents and the death of certain agents who fail to learn fast enough to satisfy survival threshold, an upper limit on classification error, P_{goal} after the t_{\max} training epochs. When the agent perishes, it is replaced with the offspring that inherits its parent’s noise intensity. Most successful agents are given a right to produce the offspring. A necessary condition to become the parent agent is its small classification error, i.e.

$$P_{\text{classif}} < \eta_{\text{child}} \times P_{\text{goal}}, \quad (\eta_{\text{child}} < 1). \quad (1)$$

We calculated P_{classif} analytically since decision boundary is a hyperplane and the data is Gaussian with known parameters. To increase a speed of genetic adjustments to

strengths of the PR tasks changes, a random variable $\epsilon \sim N(0, \sigma_{\text{mutation}}^2)$ is added to parameter α_i each time through a mutation process. Inheritance and mutations result that *during the PR task changes, values of noise intensity, α_i , are varying in time* [16].

Learning of the offspring starts from zero initial weights. The offspring is given $t_{\text{max}}^{\text{newborn}} = t_{\text{max}} + t_{\text{child}}$ training epochs before first survival test is applied.

Earlier experiments showed that small fraction of the agents were dying if environmental changes were of moderate size. In that case, we had rather slow adaptations of noise injection intensities, α_s , to alterations in the strengths of environmental changes (variation of parameter β). *To make adaptation of the noise intensity easier we need to force more agents to die.* Inspired by observations from biology and social life, in present research *we made survival threshold, P_{goal} , a random variable.* So, in each single survival test, to P_{goal} we added a random variable uniformly distributed in interval $[-\epsilon_g, \epsilon_g]$. Subsequent experiments confirmed that such strategy increased the population’s resistance to strong environmental changes.

To increase their survivability in difficult time periods, many living beings are accumulating their earnings acquired during “good times”. Therefore, we allowed the agents storing their “earnings” (a self-support chain):

$$\Delta_{\text{gain}}^{\text{new}}(j) = \Delta_{\text{gain}}^{\text{previous}}(j) \times (1 - \gamma_{\text{store}}) + (P_{\text{goal}} - P_{\text{classif}}(j)) \times \gamma_{\text{store}},$$

if survivability condition $P_{\text{classif}}(j) < P_{\text{goal}}$ was satisfied. Thus, a part of the earnings, $\Delta_{\text{gain}}^{\text{new}}(j)$, accumulated by the j^{th} agent during previous tests, are stored. A fraction γ_{gain} of this sum is used to modify the agent’s individual survivability condition

$$P_{\text{classif}}(j) < P_{\text{goal}}^* (j) = P_{\text{goal}} + \Delta_{\text{gain}}(j) \times \gamma_{\text{gain}}. \tag{2}$$

An inclusion of the self-support chain constitutes one of the novelties of this paper in comparison with previous versions of evolution models reviewed in the Introduction. Subsequent experiments have shown that accumulation of the earnings increases the ability of the agent population to adapt to severe environmental changes.

Social organizing of the agents into the groups. As shown in recent studies in the multi-agent system research, the compromises may facilitate coalition formation and increase agent utilities [6]-[12]. In this paper, we investigate rather simple cooperation model, tailored to situations where environmental changes are powerful.

Consider a strategy where most successful agents transmit their genetic code (parameter α_i) only to agents of the same group. During a sequence of most powerful environmental changes, it can happen that in some groups there are no agents that could become the parents. In such situations, the size of this group is diminishing. It can happen that all members of the group will die during long sequence of powerful changes. Therefore, in such experiments we observed extinction of the populations.

Let the most successful agents of large groups may transmit their inheritance code, parameter α_i and its group label, to an offspring of almost dying group. Simulation studies have shown that such strategy causes a death of majority of agent groups and

establishment of two or even one group in the population. In a presence of one or two groups, we observed subsequent decrease in survivability of the agent population during further environmental changes. In order to avoid destruction of the multi-group system, in this paper we considered multiagent systems (MAS) where the agents' associations to the groups were fixed *a priori*. Emulating the animal and human social behavior, in this agent society model, we assumed that only the best agents that satisfy condition

$$P_{\text{classif}}(j) < \eta_{\text{elite}} \times \eta_{\text{child}} \times P_{\text{goal}}, \quad (3)$$

are composing "an elite subgroup". In such a way, we are imitating the cooperation between the agents: the elite agents are "altruistic", i.e, they equalize their success:

$$\bar{P}_{\text{classif}}^{\text{elite}}(j) = \sum_j P_{\text{classif}}(j) / n_j^{\text{elite}}. \quad (4)$$

Therefore, the agents' excellence estimate depends on an excellence of its partners:

$$P_{\text{classif}}^{\text{elite}}(j) = (1 - \eta_{\text{altruism}}) \times P_{\text{classif}}(j) + \eta_{\text{altruism}} \times \bar{P}_{\text{classif}}^{\text{elite}}(j). \quad (5)$$

Survival conditions are not modified for the elite group. If a number of agents satisfying condition (1) in the r -th group is less than $n_{\text{aid}} = 2$, the s -th agent from other group can transfer its "genetic code", α_s , to the offspring. This agent is randomly chosen from a pool of candidate agents that satisfy condition (1). *The parent's group label, integer s , is not transmitted to offspring*, only a noise level, α_j . Altruistic strategy inside the elite group and tiny beneficial help of other groups are increasing the resistance of the agent populations to most powerful environmental changes.

Enumeration of the model's parameters

- the data dimensionality, training set size, difference in mean vectors, correlations;
- parameters of SLP training algorithm: learning step, η , targets t_1 and t_2 ;
- a character of environmental changes, most importantly, the intervals between the changes, magnitudes of the changes, the speed and other parameters of the character of variation of these magnitudes;
- survivability conditions: P_{goal} , interval $[-\varepsilon_g, \varepsilon_g]$, the numbers of epochs, t_{max} , t_{child} , allowed before survivability tests are applied to mature agents and the offspring,
- a noise injection intensities, α_i : mutations $\sigma_{\text{mutation}}^2$;
- the offspring producing condition: η_{child} ;
- the size of the agents population, m , the number of groups, K and a distribution of a number of the agents between the groups;
- self-support characteristics, γ_{store} , γ_{gain} , the helpful aid to nearly dying group, n_{aid} ;
- altruistic aid inside the elite groups, η_{elite} and η_{altruism} .

3 Simulation Experiments

Below we report results obtained with following set of the model's parameters: $m=500$, $\mu_1 = (0.15 \ 0.15)$, $t_{\text{change}}=180$, $t_{\text{max}}=120$, $t_{\text{child}}=50$, $\eta_{\text{child}}=0.8$, $P_{\text{goal}}=0.1$, $\varepsilon_g=0.02$,

$\eta=1.5, \alpha_{\min}^{\text{start}}=0.2, \alpha_{\max}^{\text{start}}=0.4, \sigma_{\text{mutation}}=0.02, n_{\text{aid}}=2, \eta_{\text{elite}}=1.2, \gamma_{\text{store}}=0.5, \gamma_{\text{gain}}=0.4$ and $\eta_{\text{altruism}}=0.25$. A dynamics of the magnitudes of 250 task changes, parameter, $\beta(s)$, is depicted in Fig. 1a. We were performing survival tests after each 10 training sweeps. Accordingly, we had 18 tests between two subsequent PR task changes.

In the experiments, starting noise injection level values, α_i , were distributed randomly in interval [0.2 0.4]. First m_α agents composed the first group ($m_\alpha=m/K$). Following m_α agents composed the second group, etc. Fig. 1b shows that more than ~33 % of the agents were dying after each survivability test if comparatively high values

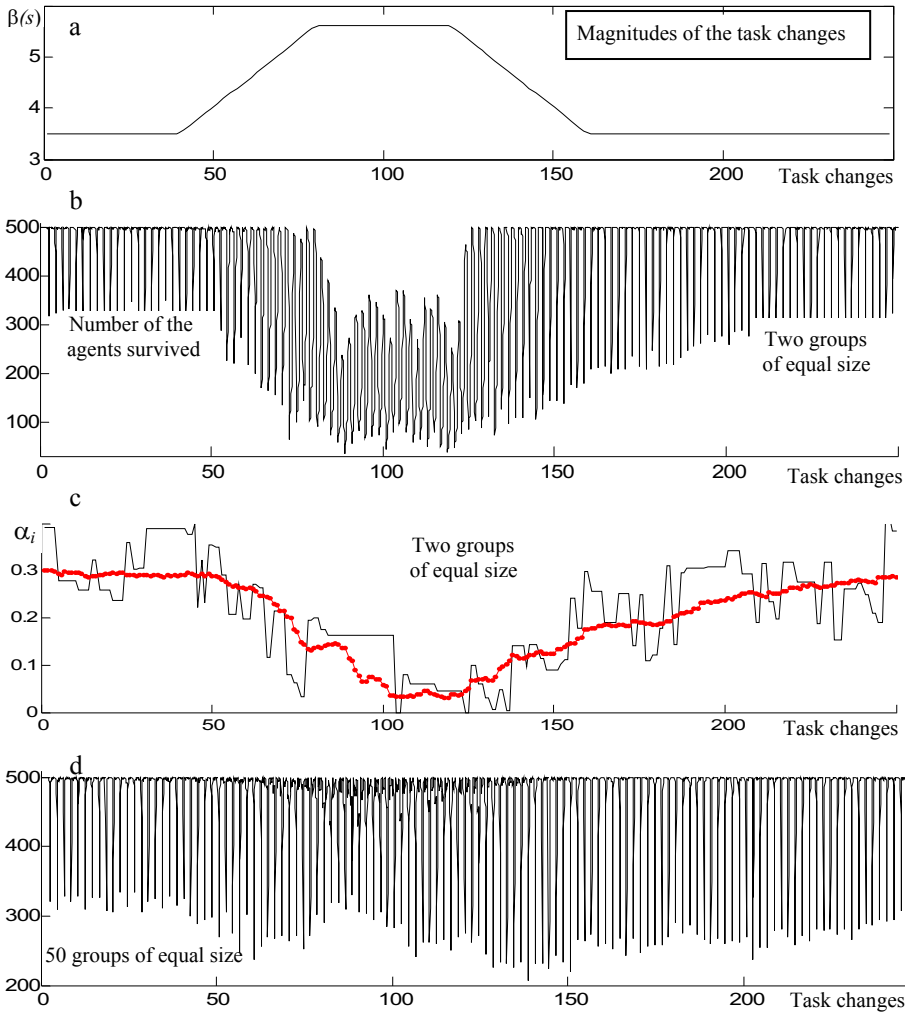


Fig. 1. Dynamics of a) - parameter $\beta(s)$, b), d) a number of surviving agents, c) a noise injection intensity (bold – a mean, thin – that of the 376th agent, α_{376})

of level of environmental changes ($\beta = 3.5$) and a noise level parameter ($\varepsilon_g=0.02$) that controls fluctuations of survivability threshold, P_{goal} , were used. Thus, after 40 task alterations the values of α_i , ($i= 1, 2, \dots, m$) adapted to strengths of the changes and made the agent population robust to the changes.

In Fig. 1c we see variation of α_i , a noise intensity parameter, during 250 changes if the agent population was split into two groups. The mean of the noise level is following the variability of the strengths of environmental changes (Fig. 1a). It is worth to stress that *in each group the agents had evolved diverse sets of parameter α_i values*. In our experiments, environmental changes were frequent ($t_{change}=180$) in comparison with maximal learning time allowed ($t_{max}=120$). Therefore, for given set of the model parameters, the strongest environmental changes ($\beta_{max}= 5.6$) caused a reduction in the noise injection intensity. In experiments with *rare environmental changes*, we observed an increase in a noise level during most powerful task changes.

The number of agents survived (Fig. 1b) is following the variation of the power of environmental changes as depicted in Fig. 1a. The minimal number of agents that survived corresponds to a period of the strongest data changes ($m_{min}=34$ agents, after the 88th task change). In a case of single agent group in the population ($K=1, m_\alpha=500$), only six agent survived, $m_{min}=6$. With an increase in the number of groups in the population, the minimal number of agents survived is increasing: in case of $K=10$ ($m_\alpha=50$), $m_{min}=90$. When $K=25$, $m_{min}=163$. When $K=50$, $m_{min}=206$. If number of groups becomes too large, $K=100$, we have smaller number, $m_{max}=196$. Thus, we observed the maximal resistance of the population to environmental changes when we had 50 groups, ten agents in each of them. This result confirms that mutual cooperation of agents inside the groups and *tiny beneficial aid to weakest groups are advantageous* for the agent population survival.

We performed also several experiments where the number of agents in one group could vary and where one group does not help to another one. After long sequences of powerful changes, small groups disappear. Only one group remains. Without outside help this group is dying too, if a sequence of the PR task changes is lengthy enough.

One more series of the experiments was performed in order to trace an influence of beneficial *altruistic cooperation of the agents* inside the groups. We have seen above, that in the population composed of two groups, $m_{max}=34$, when $\eta_{altruism}=0.25$. If $\eta_{altruism}=0.5$, we had higher resistibility of the population: $m_{max}=46$. If $\eta_{altruism}=0.75$, the resistibility increased up to $m_{max}=51$. Too high altruism of the agents, however, reduces the resistibility: for $\eta_{altruism}=0.9$, $m_{max}=14$.

Comparison of survivability dynamics graphs (Figures 1b and 1d) obtained for two population models (two groups and 50 ones) suggests that different mechanisms determine the number of agents that survived during strongest environmental changes. In both population models, approximately 33% of the agents survived if the changes were not very powerful ($\beta(s)=3.5$, the 10th – 40th data changes). During the period of the strongest data changes, almost all agents were dying in the single or two group models. In the 50 and more group models, however, the strongest environmental changes actually did not reduce the minimal number of agents. Vividly speaking, one may say that *strongest environmental catastrophes “consolidate” the grouped MAS*. This fact and other ones that follow from our simulation studies indicate that the

multi-group model of adaptive agent populations can create varieties of imitation situations and be useful in explaining a large quantity of real world observable facts.

The helpful support to nearly dying group, n_{aid} , and the aid inside the elite groups, $\eta_{altruism}$, could be interpreted as parameters that encourage altruistic aid to members to the agent own group and the agents of foreign groups. Up to now, a presence of “a gene of altruism” was not found neither in the Nature nor among mathematical models used to model population of intelligent agents [6], [8], [20], [21], [22]. We performed a number of experiments with parameter $\eta_{altruism}$, the altruistic aid inside the elite group, used as an inherited variable. If parameter $\eta_{altruism} = 0$ for all agents at the very start, later during thousands of environmental changes it is increasing steadily. This result together with above observation that the agent population with $n_{aid} = 0$ is less resistant to most powerful pattern recognition task changes suggest that, possibly, a “gene of altruism” could exist in the Nature. Therefore, the MAS model of adaptive agents acting in changing environments could become a useful instrument in such sort of analysis.

Unequal sized groups. Societal and political groups, religions and economic systems are differing in their size. For that reason, we experimented with three 400 agent systems composed of 100 unequal sized groups. In model “A” we had 35 groups of three agents, 26 groups of four agents, etc. (see histogram “A” in Fig. 2). A number of the groups of different size were the highest in model “B”. Here among 100 groups we had 63 ones composed of three agents, and one large group composed of 18 agents (histogram B in Fig. 2). In model “C” we had 25 groups composed of three agents, 50 groups of four agents and 25 groups of five agents.

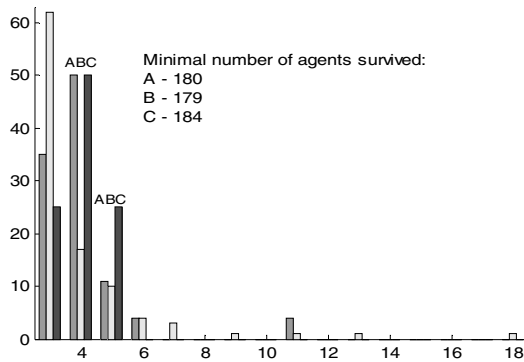


Fig. 2. A distribution of the number of the groups composed of 3, 4, up to 18 intelligent agents in three multi-group population models, A, B and C, composed of 400 agents

In all three MAS, we observed higher survivability as compared with the MAS of 100 equal-sized groups. For the same *a priori* fixed set of the model parameters and fixed value of initialization of random number generator used to generate the data, split “C” allowed minimum 184 agents to survive. This result is higher as $m_{min} = 173$ agents in equal-sized grouping. Split “A” gave $m_{max} = 180$. The “widest split” “B”, which contained only one very large group and large number of very small ones, also resulted

rather good survivability, $m_{\max} = 179$ agents. Similar results were obtained in other four series of experiments with the 400 agent systems, A, B and C.

4 Concluding Remarks

We considered the problem of solving the sequences of PR tasks in changing environments when the designer does not know a moment of subsequent task change. We analyzed multitude of pattern classifiers that compose the system of intelligent agents. In such system we have variety of adaptive pattern recognition rules that stop working if do not succeed to adapt to changes rapidly. Most successful rules are transmitting their learning style parameters to offspring classifiers. To elucidate core tendencies, we tried to use the model as simple as possible. The changes in the environments were mimicked by uncomplicated sequences of changes of the PR tasks in 2D feature space. Like in our previous papers, the learning style was controlled by degree of accidental success and misfortunes in the agent training that were mimicked as random distortions of the target patterns.

In addition to previous findings concerning usefulness of corrupted training signals introduced in order to overcome most powerful changes, in present research, we considered few novelties. We have shown that the split of the agent population into the groups of different sizes, restricted beneficial cooperation between them, forming of elite subgroups, moderate altruistic behavior of the agents inside their own elite group and continuously corrupted survivability conditions are assisting in faster adaptation of agent population to unexpected environmental changes. Our analysis paradigm gives the hints why the move from powerful environmental changes to milder ones weakens well organized agent population sometimes.

An important conclusion derived from our investigations is: such simple model as structurally organized population of *nonlinear SLPs*' trained by total gradient equipped with *dynamic changes of the PR tasks* and *corrupted training directives already explains variety of important phenomena* frequently observed in biology, social systems and modern multi-agent systems research. General conclusions obtained from our investigation paradigm could be useful for development of complex MASs of swarm intelligence and computerized studies of the social systems where instead of simple SLP based classifiers, much more complicated intelligent adaptive agents would be considered.

Our investigation has shown that modeling of adaptive agents acting in changing environments could become a useful instrument in studies of human and social altruism. In future research, more comprehensive analysis of beneficial cooperation between the agents and the agent groups could be examined.

References

1. Weng, J., McClelland, J., Pentland, A., Sporns, O., Stockman, I., Sur, M. and Thelen, E.: Autonomous mental development by robots and animals. *Science*, 291(5504) (2001) 599
2. Yao X. Evolving artificial neural networks. *Proceedings IEEE*, 87 (1999) 1423-1447
3. Hinton, G.E., Nowlan, S.J.: How learning guides evolution. *Complex Systems*, 1 (1987) 497-502

4. Nolfi, S., Floreano, D.: Learning and evolution, *Autonomous Robots*, 7 (1999) 89-113
5. Cortez, P, Rocha, M., Neves, J.: A Lamarckian approach for neural network training. *Neural Processing Letters*, 15 (2002) 105-116
6. Pepper, J., Smuts, B.: A mechanism for the evolution of altruism among non-kin: Positive assortment through environmental feedback. *Amer. Naturalist*. 160 (2002) 205-213
7. Fehr, E., Fischbacher, U.: The nature of human altruism. *Nature*, 425 (2003) 787-791
8. Gintis, H., Bowles, S., Boyd, S., Fehr, E.: Explaining altruistic behavior in humans. *Evolution and Human Behavior* 24 (2003)153–172
9. Panait, L., Luke, S, Wiegand, R.P.: Biasing coevolutionary search for optimal multi-agent behaviors. *IEEE Trans. on Evolutionary Computation*, 10 (2006) 629-645
10. Ferber, J.: *Multi-Agent Systems: Towards a Collective Intelligence.*: Addison-Wesley, Reading, MA (1998)
11. Yakoo, M.: *Distributed Constraint Satisfaction: Foundations of Cooperation in Multi-Agent Systems*, Springer-Verlag, Berlin Heidelberg New York (2001)
12. Iwata, K., Miyazaki, M., Ito, N., Ishii, N.: Increasing the efficiency of cooperation among agents by sharing actions, *Lecture Notes in Computer Science*, Vol. 3026. Springer-Verlag, Berlin Heidelberg New York (2004) 279-289
13. Raudys S: An adaptation model for simulation of aging process. *Int. J. of Modern Physics, C*. 13 (2002) 1075-1086
14. Raudys S. and Justickis V.: Yerkes-Dodson law in agents' training. *Lecture Notes in Artificial Intelligence*, Vol. 2902. Springer-Verlag, Berlin Heidelberg New York (2003) 54-58
15. Raudys, S.: Effect of synthetic emotions on agents' learning speed and their survivability. *Lecture Notes in Artificial Intelligence*, Vol. 3630. Springer-Verlag, Berlin Heidelberg New York (2003) 1 – 10
16. Raudys, S: Survival of intelligent agents in changing environments. *Lecture Notes in Artificial Intelligence*, Vol. 3070. Springer-Verlag, Berlin Heidelberg New York (2004) 109-117
17. Raudys, S.: On the universality of the single-layer perceptron model. In: Rutkowski L. (ed.): *Neural Networks and Soft Computing Physica-Verlag (Springer)*, Berlin Heidelberg New York (2002) 79-86
18. Haykin, S.: *Neural Networks: A comprehensive foundation*. 2nd edition. Prentice-Hall, Englewood Cliffs, NJ (1999)
19. Raudys, S.: *Statistical and Neural Classifiers: An integrated approach to design*. Springer-Verlag, London Berlin Heidelberg (2001)
20. Boyd, R.: The puzzle of human sociality. *Science* 314 (8 Dec. 2006) 1555-1556
21. Nowak, M.A.: Five rules for the evolution of cooperation. *Science* 314 (8 Dec. 2006) 1562-1563
22. Bowles, S.: Group competition, reproductive leveling, and the evolution of human altruism. *Science* 314 (8 Dec. 2006) 1569-1572

Softening Splits in Decision Trees Using Simulated Annealing

Jakub Dvořák and Petr Savický

Institute of Computer Science, Academy of Sciences of the Czech Republic,
Prague, Czech Republic

`dvorak@cs.cas.cz`, `savicky@cs.cas.cz`

Abstract. Predictions computed by a classification tree are usually constant on axis-parallel hyperrectangles corresponding to the leaves and have strict jumps on their boundaries. Frequently a better approximation may be expected, if the prediction function of the original tree is replaced by a continuous approximation. The approximation is constructed using the same training data on which the original tree was grown and the structure of the tree is preserved.

The current paper uses the model of trees with soft splits suggested by Quinlan and implemented in C4.5, however, the training algorithm is substantially different. The method uses simulated annealing, so it is quite computationally expensive. However, numerical test with data derived from an experiment in particle physics shows that besides the expected better approximation of the training data, also smaller generalization error is achieved.

1 Introduction

Classification trees are suitable for predicting the class in complex distributions, if a large sample from the distribution is available. The classical parametrical methods may not succeed in such situations, if they work with a closed formula describing the density in the whole predictor space. Decision trees and ensembles of trees are comparable to neural networks and SVM in classification accuracy. The predictor vector for a tree consists of a fixed number of numerical and categorical variables. In this paper, we consider single univariate decision trees with numerical predictors.

A trained classification tree usually does not only provide a discrete classification, but also an estimate of the confidence for it on a continuous scale. This confidence may be an estimate of the conditional probability of the classes, but this is not necessary. Even if it is not a good estimate of the probabilities, it may be a reasonable information. In the real world problems, it is frequently plausible to assume that the function which assigns such a confidence to each point of the predictor space is continuous. Even if the true distribution has a sharp boundary between the classes, a limited sample from the distribution does not provide enough information for a justified construction of a prediction function with a strict jump.

Classification trees constructed using the traditional methods like CART [1] or C4.5 [2] generate trees, whose internal nodes contain conditions of the form $x_{k_j} \leq c_j$, where x_{k_j} is one of the predictors and c_j is a threshold value. The result of the use of such sharp threshold conditions is that the predictions computed by a classification tree are constant on axis-parallel hyperrectangles corresponding to the leaves and have strict jumps on their boundaries. Such functions may badly approximate boundaries of different type. This is the cost, which is paid for the simplicity of the classifier.

A soft threshold condition means that if the actual value of x_{k_j} is close to c_j , then both branches of the tree are evaluated and their results are combined using weights changing continuously with $x_{k_j} - c_j$. Soft splits were suggested first by Quinlan [2] including the implementation in C4.5. We use exactly the same extension of the decision tree classifiers, but use a substantially different technique for training.

Quinlan's technique determines the soft thresholds in each node separately using statistical estimation. In our approach, several thresholds corresponding to a group of nodes close to each other in the tree are adjusted simultaneously. Hence, the choice of the final thresholds is influenced also by the interactions between predictors. A nonsoft tree together with the training data used for its construction, are used as the input to an optimization phase, which tries to find the values of the parameters of the soft thresholds, which yield the best possible approximation of the training data. Since the structure of the tree, in particular, its number of nodes, is fixed during the optimization, overfitting was not observed in our experiments, although a computationally intensive optimization is used to tune the soft thresholds.

The goal of the postprocessing of the tree is to reach a smoother function that fits better the training data. Since the complexity of the classifier does not increase too much, one may expect to achieve also smaller generalization error. Besides better approximation in cases, where the unknown conditional probability function is continuous, we may obtain a better approximation even if the true value of the conditional probability makes a jump on a boundary between the regions of different classification, if the boundary is not in axis-parallel direction. Soft tree may represent a more general function than a nonsoft tree. In particular, as a consequence of interactions of several predictors, the prediction function of a soft tree may have gradient vector in a general direction, while keeping the small number of nodes of the original tree. Approximating this using a nonsoft tree requires to use a stair like boundary with a large number of nodes.

The current paper investigates classification accuracy on data from particle physics (MAGIC gamma telescope¹) considered already in [3]. In this comparison, ensembles of trees, in particular random forests, provided the best classifiers for these data. Our experimental results on these data demonstrate that the trees obtained by introducing soft splits may have substantially smaller generalization error than individual nonsoft trees.

¹ <http://wwwmagic.mppmu.mpg.de>

We also compare the soft trees obtained by simulated annealing with soft trees obtained by C5.0². There is no significant difference in test classification error between these two types of trees, see section Results, although the trees are obtained using substantially different principles. C5.0 finds the soft thresholds by estimating the uncertainty in determining the cut value. This is done by finding two shifted cut values, one smaller and one larger than the original cut value, so that the number of errors on the training set in the node increases by one standard deviation, when using the shifted cut points, see [2]. This is done in each node separately and frequently increases the training error. On the other hand, our approach optimizes the soft threshold purely by minimizing the training error. Our result shows that minimizing training error leads to similar results as the statistical estimation used in C5.0, at least on the MAGIC data.

A different approach is described in [4], which directly builds a soft decision tree using local optimization in individual nodes. Building a fuzzy tree defined using fuzzy set operations is presented in [5].

2 Decision Tree with Soft Splits

Let T be a decision tree with nodes v_j for $j = 1, \dots, s$. We assume that if v_{j_1} and v_{j_2} are left and right successor of v_j , then $j < j_1$ and $j < j_2$. In particular, v_1 is the root. Let $\text{Splits}(T)$ be the set of indices of the internal nodes and $\text{Leaves}(T)$ the set of indices of the leaves. The variable tested in node v_j is denoted x_{k_j} and the corresponding threshold value is denoted c_j . If C is the number of classes, then the label (response vector) $G(v)$ of a leaf v is a nonnegative real valued vector of dimension C , whose coordinates sum up to one. Let $T(x)$ be the function $\mathbb{R}^d \rightarrow \mathbb{R}^C$ computed by the tree, where d is the number of predictors. More generally, let $T_j(x)$ be the function computed by the subtree starting at v_j . In particular, $T_1(x) = T(x)$. Note that $T_j(x)$ is defined even in cases, when the computation of the whole tree T for x does not reach the node v_j .

If v_{j_1} and v_{j_2} are left and right successor of v_j , then we have $T_j(x) = \text{if } x_{k_j} \leq c_j \text{ then } T_{j_1}(x) \text{ else } T_{j_2}(x)$. If we define $I(\text{condition})$ equal to 1 if *condition* is true and equal to 0 if *condition* is false, then this is equivalent to

$$T_j(x) = I(x_{k_j} \leq c_j)T_{j_1}(x) + I(x_{k_j} > c_j)T_{j_2}(x).$$

A tree with soft splits is obtained by replacing this by

$$T_j(x) = L_j(x_{k_j} - c_j)T_{j_1}(x) + R_j(x_{k_j} - c_j)T_{j_2}(x)$$

for appropriate continuous functions $L_j, R_j : \mathbb{R} \rightarrow [0, 1]$. It is required that if both subtrees of T_j return the same output vector, then T_j returns the same vector as well. Hence, we require $L_j(t) + R_j(t) = 1$ for all $t \in \mathbb{R}$. A natural further requirement is that L_j be nonincreasing with limits $L_j(-\infty) = 1$ and $L_j(\infty) = 0$. Hence, we also have that R_j is nondecreasing with limits $R_j(-\infty) = 0$ and $R_j(\infty) = 1$.

² <http://www.rulequest.com>, commercial version of C4.5.

The functions L_j and R_j used in the current paper are piecewise linear functions interpolating the points in the table

| | | |
|-----------|----------|----------|
| t | $L_j(t)$ | $R_j(t)$ |
| $-\infty$ | 1 | 0 |
| $-a_j$ | 1 | 0 |
| 0 | 1/2 | 1/2 |
| b_j | 0 | 1 |
| ∞ | 0 | 1 |

where the values $a_j \geq 0$ and $b_j \geq 0$ are parameters of the soft splits. If $a_j > 0$ and $b_j > 0$, then the functions L_j, R_j are uniquely determined by the above table. If some of the values a_j, b_j is zero, the corresponding function L_j, R_j is defined as a pointwise limit, which is noncontinuous. The limit function satisfies $L_j(0) = 1/2$ or $R_j(0) = 1/2$ as in any other case.

The tree with soft splits is obtained by replacing the evaluation function in each internal node by the above one. This requires to specify the parameters (a_j, b_j) in all internal nodes. Let $\theta = \{(a_j, b_j)\}_{j \in \text{Splits}(T)}$. The functions L_j, R_j defined above depend on θ . So, the correct notation for them is $L_j(\theta, t), R_j(\theta, t)$. Moreover, let $T(\theta, x)$ and $T_j(\theta, x)$ denote the functions computed by the whole tree and the tree starting at node v_j , respectively, if the soft splits determined by θ are used. We again have $T(\theta, x) = T_1(\theta, x)$.

If $x_{k_j} = c_j$, then the soft split always evaluates both subtrees and returns their arithmetic mean. If $x_{k_j} \neq c_j$ the situation depends on x_{k_j} as follows. If $x_{k_j} \leq c_j - a_j$ or $x_{k_j} \geq c_j + b_j$, then the evaluation function in node v_j behaves in the same way as in the original tree and returns the value of one of the two subtrees. If $x_{k_j} \in (c_j - a_j, c_j + b_j)$, then evaluating $T_j(x, \theta)$ requires to compute both subtrees and the output is a combination of their values.

Note that $T(0, x)$ behaves similarly to $T(x)$, but there is a difference. If the computation of $T(x)$ never reaches a node, where $x_{k_j} = c_j$, then we have $T(0, x) = T(x)$. However, if $x_{k_j} = c_j$ is satisfied at some step of the computation, the results may differ, since evaluation of $T(0, x)$ combines both subtrees, while evaluation of $T(x)$ uses only one of them.

For every pair of nodes v_j and v_{j_1} such that v_{j_1} is one of the two successors of v_j , let $H(\theta, v_j, v_{j_1})(x)$ be the weight, with which the subtree in v_{j_1} is considered, when evaluating $T_j(\theta, x)$. We have

$$H(\theta, v_j, v_{j_1})(x) = \begin{cases} L_j(\theta, x_{k_j}) & \text{if } v_{j_1} \text{ is the left successor of } v_j \\ R_j(\theta, x_{k_j}) & \text{if } v_{j_1} \text{ is the right successor of } v_j \end{cases}$$

For every leaf v , let $\text{Path}(v)$ be the uniquely determined path from the root to v . Then an explicit formula for the function computed by a tree with soft splits is

$$T(\theta, x) = \sum_{\substack{j \in \text{Leaves}(T) \\ (u_1, \dots, u_k) = \text{Path}(v_j)}} G(v_j) \prod_{i=2}^k H(\theta, u_{i-1}, u_i)(x).$$

The formula may be verified by induction starting at the leaves.

3 Optimization of the Soft Splits

The method described in this paper assumes that a nonsoft classification tree T with two classes 0 and 1 is available. Such a tree may be obtained using a method like CART or C4.5. We used the R implementation of CART. The response vector is two dimensional in this case. The two coordinates represent confidence for classification into each of the classes and are assumed to sum up to one. Hence, each of the coordinates alone carries the full information on the prediction. Let us denote $T^*(x)$ the component of the response vector computed by tree T , which corresponds to class 1. This may be used for classification by choosing a threshold h and using the rule “predict 1 iff $T^*(x) \geq h$ ”.

The goal is to find θ such that the error of classification of the softened tree $T(\theta, x)$ on unseen cases is smaller than in the original tree. The same threshold h and the same rule $T^*(\theta, x) \geq h$ for classification into class 1 is used as for the original nonsoft tree. Since the algorithm has access only to the training data, the method tries to achieve the above goal by minimizing the error of $T^*(\theta, x)$ on the training data. This error may be measured in different ways. We tested first simply the classification error, but it appeared to be better to use a continuous error defined on the data (x_i, y_i) , $i = 1, \dots, m$ by the following formula

$$f(\theta) = \sum_{i=1}^m e^{\alpha(|T^*(\theta, x_i) - y_i| - 1)}, \quad (1)$$

where α was chosen to be 4.

Since the minimized function $f(\theta)$ is not a smooth function and has a large number of local minima, we used simulated annealing available in R Statistical Package [6] using method SANN of function “optim”. Since this does not allow to restrict the range of θ to nonnegative values, we used a large penalty (number of errors larger than the number of training cases) for θ , which contain a negative value. The initial value of θ was chosen to be 0, i.e. the optimization starts approximately at the original nonsoft tree.

The dimension of the optimization problem (the number of parameters of the minimized function) is two times the number of internal nodes. In order to make the optimization process independent on the scaling of the data, the optimization function uses a normalized vector of parameters $\theta' = \{(a'_j, b'_j)\}_{j \in \text{Splits}(T)}$, where $a'_j = a_j/a_{j,0}$, $b'_j = b_j/b_{j,0}$. The normalizing factors $a_{j,0}, b_{j,0}$ are defined using the original nonsoft tree.

It appeared to be better to split the minimization into phases, in each of which, only a small randomly chosen subset of arguments is modified. The arguments are selected so that they correspond to edges close in the tree. The maximum number of simultaneously considered edges was 7.

Optimization is performed by a sequence of calls of method SANN of optim. The initial approximation of each call is the best solution found during the previous call. The initial temperature for all calls is $temp = 10$ and the bound on the number of iterations is $maxit = 101$ for all calls. One call of optim is successful, if it succeeds to find a better solution than the initial one. The whole process stops, when 50 consecutive calls are unsuccessful.

4 Experimental Setup

We performed 20 runs of the experiment. In each single run of the experiment, the available data D were split at random in ratio 2:1 into a training set D_1 and a test set D_2 and the four classifiers obtained by the following methods were constructed:

1. CART.
2. Soft tree obtained by the method described in the previous section from CART trees.
3. C5.0 without softening.
4. C5.0 with softening (option “-p”).

More detail is given in subsections below.

4.1 CART

The training set D_1 was further split in ratio 2:1 into D_{11} and D_{12} . The larger part D_{11} was used for growing the tree, the smaller D_{12} was used for pruning as the validation set (cost complexity pruning in CART method). The result of the pruning is a sequence of trees of different sizes. For our experiment, we used only the subsequence between the first tree with at least two leaves and the standard output tree of CART method, which is the one with the smallest error on the validation set. Accuracy of the trees in this subsequence is reported for comparison with the other methods. In order to show that the comparison result does not depend on the selection of the tree in the subsequence (e.g. 1st rule), we select the best tree on the test set D_2 and report its accuracy. Even such trees are worse than the soft trees, whose error is measured using standard methodology.

4.2 Softening Trees from CART

We use the subsequence of pruned trees constructed by CART as described above. The parameters of the soft splits in these trees were optimized as described in Section 3. When interpreting the soft tree as a classifier, we used threshold $h = 0.5$. This means that the class with the larger confidence (we have two classes) in the response vector is predicted.

The error of the resulting soft tree is never worse than in the original tree, however, it is sometimes close to it. Such soft trees are discarded. The optimization is considered unsuccessful, if the ratio of the error of the original tree over the error of the soft tree is less than 1.01. The sequence of trees from CART contains trees of different sizes. Smaller trees have higher chance to be improved by one run of the softening procedure. On the other hand, if the softening procedure succeeds to improve a large tree, the result is usually better than for small trees. In order to balance between these two effects, we used a strategy which is splitted into steps numbered by $i = 1, 2, \dots$. In step i , the softening procedure tries to improve all of the i largest trees in the sequence. The process

terminates, when 10 trees successfully improved by softening are collected. The resulting classifier is determined as the tree with the smallest classification error on D_1 among the 10 trees obtained by the above strategy. The error of this tree on D_2 is reported.

4.3 C5.0

We used C5.0 release 1.15. The confidence level, which determines the amount of pruning was chosen 0.01 (option “-c 1”), which appeared to be the best among several values of the confidence level between 0.003 and 0.1 in a geometric sequence, which we tried. The chosen value lead to the smallest median of the error for the 20 tested splits of the data. For each split of the data, C5.0 was run twice, with and without the option “-p”, which forces that a softened tree is constructed.

5 Results

For our experiments, we used data simulating registration of gamma and hadron particles in Cherenkov imaging gamma ray telescope MAGIC [3]. There are 10 numerical predictors and 2 classes. The predictors are numerical values that are produced by the registration device and characterize the registered particle. Class signal represents cases, where the registered particle is gamma. Class background corresponds to hadrons, mostly protons. The number of cases in the dataset is 19020.

The data were created by a complex Monte Carlo simulation [7] that approximates the development of a shower of particles generated by a high energy primary particle that reaches the atmosphere. The result of the simulation is an estimate of the number of Cherenkov ultraviolet photons that reach different pixels in the focus of an antenna at the ground and form a single registered event. The 10 predictors are numerical parameters of the geometric form of the obtained image.

We used 20 random splits of this set in the ratio 2:1 into D_1 and D_2 . For each of these splits, four classifiers were constructed using the methods described in the previous section. The classifier was constructed using D_1 and its accuracy was estimated using D_2 . As mentioned in Section 4.1, the accuracy of nonsoft CART is the best result among the trees in the considered subsequence obtained by pruning.

The table in Fig 1 summarizes the test errors on D_2 for all four used methods. The table demonstrates that both types of softening improve accuracy with respect to the corresponding nonsoft version of the algorithm. Moreover, the results of the two methods of softening are close to each other. In the following two paragraphs, we support these two conclusions by further numerical evidence.

The error of the soft tree obtained by simulated annealing was always by at least 10% smaller (relatively) than that of the corresponding nonsoft tree. In order to formulate the result in terms of statistical significance, let p be the

| | CART non-soft | CART softened | C5.0 non-soft | C5.0 softened |
|--------|------------------|------------------|------------------|------------------|
| mean | 0.1578 | 0.1380 | 0.1488 | 0.1395 |
| sd | 0.0052 | 0.0042 | 0.0033 | 0.0028 |
| median | 0.1560 | 0.1381 | 0.1489 | 0.1393 |

Fig. 1. The classification error on the test set for the four considered methods

probability of the event that the error rate of a tree softened using simulated annealing on the MAGIC dataset is lower than 90 % of error rate of CART tree. The one-sided lower 0.95 confidence limit for p obtained using the exact binomial test is approximately 0.86, since we have 20 successes out of 20 trials.

In order to verify that softening using simulated annealing led to soft trees with similar accuracy on the MAGIC dataset compared to that of C5.0 softened tree, we use two sided Wilcoxon signed-rank test. The results of simulated annealing are slightly better, but after removing one zero difference, the p -value of the test was 0.2684 confirming that the difference in errors of the two methods is not significant.

| | CART non-soft | CART softened | C5.0 non-soft | C5.0 softened |
|--------|------------------|------------------|------------------|------------------|
| mean | 0.1410 | 0.1301 | 0.1141 | 0.1215 |
| sd | 0.0054 | 0.0035 | 0.0048 | 0.0037 |
| median | 0.1430 | 0.1301 | 0.1142 | 0.1218 |

Fig. 2. The classification error on the training set for the four considered methods

The fact that the simulating annealing approach produces results competitive to C4.5 softening method is interesting, since there is a significant difference between the two methods. In order to demonstrate this, we present the error of the four methods on the training set, see Fig 2. The error of nonsoft CART trees on the training set is larger than the error of nonsoft trees from C5.0, which is possibly due to the fact that C5.0 required larger trees to achieve the best accuracy in our experiment. However, for CART trees, softening using simulated annealing decreases the error on the training set, while softening of the trees from C5.0 increases the error on the training set. The average ratio of the error of softened tree over the error of nonsoft tree was 0.927 for simulated annealing and 1.066 for C5.0.

6 Conclusion

Softening thresholds in decision trees using simulated annealing is competitive to the method used in C4.5 and C5.0. This shows that minimization of the error on the training data is a promising alternative approach for softening. The difference between the two methods is demonstrated by the fact that in

our experiment they change the training error in opposite directions. Our technique is based on minimization of the training error, while the original approach frequently increases the training error.

Acknowledgement. The first author was supported by Czech Science Foundation (GACR) under the grant No. GD201/05/H014. The second author was partially supported by the “Information Society” project 1ET100300517. Both authors were partially supported also by the Institutional Research Plan AV0Z10300504.

References

1. L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, *Classification and Regression Trees*, Belmont CA: Wadsworth, 1993.
2. J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo — California, 1993.
3. R.K. Bock, A. Chilingarian, M. Gaug, F. Hakl, T. Hengstebeck, M. Jiřina, J. Klaschka, E. Kotrč, P. Savický, S. Towers, A. Vaicilius, “Methods for multidimensional event classification: a case study using images from a Cherenkov gamma-ray telescope.” *Nuclear Instruments and Methods in Physics Research, Section A*, Vol. 516, Issue 2-3, pp. 511-528, 2004.
4. C. Olaru and L. Wehenkel, “A complete fuzzy decision tree technique”, *Fuzzy Sets and Systems*, Vol. 138, pp. 221-254, 2003.
5. C. Z. Janikow, “Fuzzy Decision Trees: Issues and Methods”, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 28, Issue 1, pp. 1-14, 1998.
6. R Development Core Team (2005). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. URL <http://www.r-project.org>.
7. D.Heck et al., CORSIKA, *A Monte Carlo code to simulate extensive air showers*, Forschungszentrum Karlsruhe FZKA 6019, 1998.

A Novel Architecture for the Classification and Visualization of Sequential Data

Jorge Couchet¹, Enrique Ferreira², André Fonseca³, and Daniel Manrique⁴

¹ Universidad ORT, 11100 Cuareim 1451, Montevideo, Uruguay
jorge.couchet@universidad.ort.edu.uy

² Universidad Católica del Uruguay, 11600 8 de Octubre 2738, Montevideo, Uruguay
enferrei@ucu.edu.uy

³ Universidad ORT, 11100 Cuareim 1451, Montevideo, Uruguay
fonseca@ort.edu.uy

⁴ Universidad Politécnica de Madrid, 28660 Boadilla del Monte, Madrid, Spain
dmanrique@fi.upm.es

Abstract. This paper introduces a novel architecture to efficiently code in a self-organized manner, data from sequences or a hierarchy of sequences. The main objective of the architecture proposed is to achieve an inductive model of the sequential data through a learning algorithm in a finite vector space with generalization and prediction properties improved by the compression process. The architecture consists of a hierarchy of recurrent self-organized maps with emergence which performs a fractal codification of the sequences. An adaptive outlier detection algorithm is used to automatically extract the emergent properties of the maps. A visualization technique to help the analysis and interpretation of data is also developed. Experiments and results for the architecture are shown for an anomaly intrusion detection problem.

1 Introduction

A proper processing of series of sequential data involves the resolution of non trivial problems related to the interaction and interdependence between the elements of a sequence. In particular, we can outline the following issues:

- the design of mechanisms to capture the sequential correlation (spatial or temporal) within the data,
- the development of strategies to identify and capture long-term dependences between the different attributes of a sequence,
- the design of computationally efficient algorithms to implement the mechanisms that solve the previous issues.

A powerful tool to organize, study and capture the properties of a complex system is the analysis of the relationship between its elements, parts and the whole system through the modelling of its emergent properties. Emergence appears in a system through the properties of the collective behavior of the elements which

¹ This research is being funded by the Spanish Ministry of Science and Education under project ref. DEP2005-00232-C03-03.

the system is composed of. Looking at the system as a whole, a global emergent property comes as a new entity of the interdependency of its parts [2].

Considering a sequence of data as a system made of elements and interdependent parts, obtaining its emergent properties becomes a useful tool to capture and analyze the sequential correlation between the elements of the sequence, in particular, long-distance interactions.

Self-organizing maps (SOM) [3] are a common tool used to capture emergent properties as in Emergent Self-Organizing Maps (ESOM) [4], [5]. However, SOMs work with vectors in a finite dimension space so that, they do not manage sequential data directly. An alternative to overcome this problem is by preprocessing, e.g. with sliding windows. Such approaches are usually time consuming, bringing about the “curse of dimensionality”, and more important, the loss of information. There have been several attempts to extend the use of SOM to sequential data. The more relevant approaches are: *Temporal Kohonen Map* (TKM), *Recurrent Self-Organizing Map* (RSOM), *Recursive SOM* (RecSOM), *SOM for Structured Data* (SOMSD) and *Merge SOM* (MSOM) [6], [7], [8], [9], [10]. We observe that MSOM is either computationally more efficient, more flexible, or has theoretical properties more attractive than its predecessors [10]. A MSOM captures sequence dynamics through an internal distributed representation called *context*, that stores activation profiles of recursive substructures in the map. Each unit of the MSOM has two associated vectors, a weight vector like a SOM, and a context vector. The context information is learned also by self-organization using a reference to the winner unit of the previous element of the sequence. This recursive reference is defined as a linear combination of the weight and context vectors of the previous winner. The contribution of the current and previous elements are controlled separately. It has been shown that the recursive representation explained leads to an efficient fractal encoding of the sequence [11]. After training, the context vector converges to the fractal encoding of the elements of the current sequence. It is important to note that the optimal context vectors are stable fixed points of the MSOM dynamics (unlike the TKM case).

The present article extends the concept of ESOM to add recurrence as in MSOM [10], obtaining a R-ESOM. To capture automatically the emergent properties in the ESOM, a Local Outlier Factor algorithm (LOF) [12] is implemented. The structure introduced is called LR-ESOM. The LOF algorithm is also used to introduce a new method to obtain a visual representation of the map, called *L-Matrix*. Finally, the *LR-ESOM* and *L-Matrix* are combined in a new architecture to achieve an inductive model for sequential data along with a visualization method to help analyze and understand the sequences.

2 Proposed Architecture

2.1 Local Outlier Factor

The Local Outlier Factor (LOF) algorithm finds outliers in a multidimensional dataset [12]. A LOF value is defined for each object in the dataset being studied.

It is called local since only a restricted neighborhood of the object is used to compute it. This approach makes it possible to adapt dynamically in the presence of clusters with different densities as opposed to other global methods. To compute the LOF the parameter k has to be chosen, which is taken as the number of nearest neighbors used to define the local neighborhood of an object. For each object, its LOF is not a binary value. Instead, it represents the degree by which that object can be considered an outlier. A theoretical property is that a LOF value close to 1 means the object is deep inside the cluster.

2.2 Recurrent Emergent Self-Organizing Maps with LOF (LR-ESOM)

Emergence through self organization is a non trivial property of SOMs [4]. For emergence to appear, the system should consist of simple but highly correlated elements. Without these correlations, emergence is not possible. A model with sufficiently rich correlation and order should give global emergence [2]. A SOM with a small number of units performs like a K-means algorithm with K equal to the number of units in the map [13]. In this cases, the topology preservation property of the SOM gives little advantage. Emergence needs a *large* number of units, usually in the order of thousands of units. These maps are called Emergent SOMs (ESOM). The considered maps should also have the properties of *borderless* (to avoid border effects) and *rectangular topology* (to reduce projection errors) [5].

In this paper we introduce R-ESOM, an extension of ESOM to deal with sequential data. R-ESOM adds an explicit context representation, which modifies the distance function and learning algorithms according to [10]. Besides the weight vector $w^i \in R^d$, a context vector $c^i \in R^d$ is added. Given a sequence input x^t , the best matching unit (BMU) I^t is defined as the neuron i such that its recursive distance,

$$d_i(t) = (1 - \alpha)\|x^t - w^i\|^2 + \alpha\|c^t - c^i\|^2, \quad 0 \leq \alpha \leq 1, \quad (1)$$

to the current input and context is minimum. The descriptor context is defined as

$$c^t = (1 - \beta)w^{I^{t-1}} + \beta c^{I^{t-1}}, \quad 0 \leq \beta \leq 1, \quad (2)$$

a convex combination of the vectors associated to the BMU of the previous input of the sequence. A typical value for the merging parameter β is 0.5. The parameter α is adapted through an entropy driven algorithm, with a small initial value (e.g. 0.01).

The training algorithm adapts weight and context vectors of all the units towards the current input and context values, weighted by a neighborhood function $h(\cdot)$ of the distance d of the units to the current BMU.

$$\Delta w^i = \gamma_1 h(d(i, I^t))(x^t - w^i), \quad \Delta c^i = \gamma_2 h(d(i, I^t))(c^t - c^i), \quad (3)$$

where γ_1, γ_2 are the learning rates, usually equal. In this way, the map is trained with all the subsequences of each sequence of the training set, helping to capture long-term dependences.

Finally, the R-ESOM network is extended to a LR-ESOM using the LOF algorithm to capture emergent properties automatically. Once trained, the LOF values for each unit in the network are computed using only the weight vectors because the data sequences do not have context vectors. However, the context information is implicitly used through the training procedure. When a sequence is classified or used for training, its LOF is computed using the recursive distance (II) with context explicit information plus the precalculated LOF values for map units. For a sequence, the following values are stored:

- *Last*: the LOF of the last sequence element with respect to the units of the map,
- *Lavg*: the mean value of the LOF through the sequence, i.e. the mean value of the LOF of all sequence elements with respect to the units of the map.

2.3 L-Matrix

The development of a visual tool for the LR-ESOM is an important component to recognize the structures generated through learning, because humans have a remarkable capacity to detect 3D patterns in an image. The *L-Matrix* concept developed associates to every unit in the LR-ESOM map its LOF value.

The 3D representation of the *L-Matrix* assigns coordinates (x, y, z) to each unit, where (x, y) are the position of the unit in the map and z its LOF. The cloud of points obtained is interpolated using Bi-Cubic Bezier Patches [14] to generate a smooth surface. In order to take advantage of the LOF theoretical properties, the surface is colored, assigning the same color to units with LOF inside $[0, L_{INF}]$, with $L_{INF} = 1$. The color for units with LOF higher than L_{INF} is calculated using a mapping between a color scale and the range $(L_{INF}, L_{MAX}]$, where L_{MAX} is the maximum LOF of the units. Changing L_{INF} may be used as a zoom in/out function of the emergent properties. In our case LOF values larger than 1.5 can be considered outliers.

2.4 Architecture: Hierarchical Emergent Compression Machine (H-ECM)

The architecture proposed in this work consists of a hierarchy of N emergent compression machines (ECM) as shown in Fig. II.

The main idea behind the ECM is the use of the emergent properties for information compression. The key structure in the ECM is an emergence self-organizing map with LOF. If the input data comes from a sequence, a recurrent ESOM with LOF is used (LR-ESOM), otherwise an ESOM with LOF is used (L-ESOM). After training the ECM, given a new input S , its BMU can be found (BS) and a compact codification for S is defined as $O = \langle C, L1, L2, L3 \rangle$, where

- C is the codification of the position of BS in the map. The context associated to BS may be added for more complex sequential data,
- $L1$ is the LOF associated to BS ,

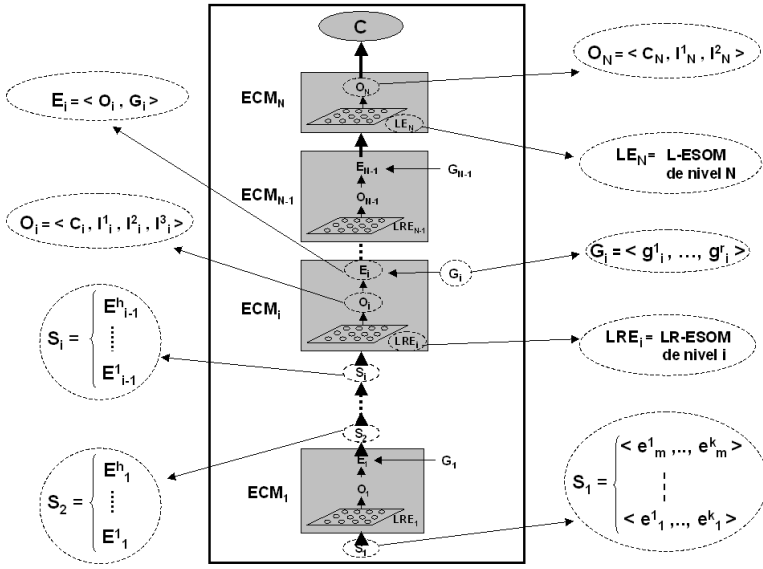


Fig. 1. Architecture proposed

- if the map is a LR-ESOM, $L2$ is the *Last* value of S , otherwise it is the LOF value of S with respect to the map,
- if the map is a LR-ESOM, $L3$ is the *Lavg* value of S , otherwise is null.

When the ECM map is a LR-ESOM, the O vector may be augmented with a global attribute vector G , generating the vector $E = \langle O, G \rangle$. The components of G represent those attributes of the sequence S that are independent of the sequential correlation (spatial or temporal) to be captured by the map.

The number of ECM levels to use depends whether the input S is a simple sequence or has nested sequences. In the case of a simple sequence, the architecture has two levels. The first level consists in a LR-ESOM map that compacts the sequence. The second level is a L-ESOM map to capture the emergent properties of the sequence taking advantage of the compact representation developed in the first level and the global attributes added at this level.

When the input S is a sequence of sequences, e.g. in the case of biometrics data, the architecture should have as many levels of ECMs with a LR-ESOM map as there are levels of sequences. These levels are followed by a level using a L-ESOM to capture the global emergent properties of the last LR-ESOM as in the simple case.

The final level of the H-ECM architecture is a supervised learning algorithm (C) to generate an inductive model of the data given. The type of supervised learning scheme depends mainly on the application characteristics. We suggest a Support Vector Machine (SVM) scheme because of its generalization properties [15]. Each ECM in the architecture has an associated L -Matrix to be able to visualize the emergent properties at all levels, helping in the interpretation of the results.

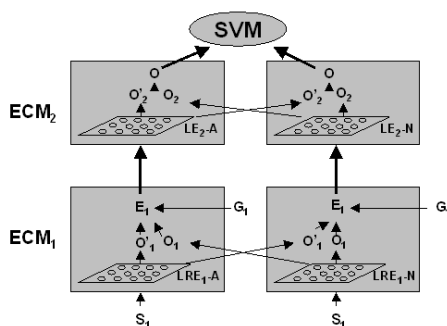


Fig. 2. H-ECM architecture for IDS

3 Architecture Evaluation

A benchmarking problem is used to evaluate the H-ECM architecture: the *Intrusion Detection* problem. In this problem the goal is to detect in the sequence of TCP/IP packets, the ones which may be harmful to the system. A set of TCP/IP connections of normal traffic (*NT*) and attack traffic (*AT*) are provided in the Intrusion Detection DARPA 1998 problem set [2]. For this experiment, HTTP connections are used. The connections are modeled as simple sequences of TCP/IP packets. Therefore, the H-ECM applied has two levels of ECM (a LR-ESOM and a L-ESOM) plus the supervised SVM machine for the final classification. In order to obtain a better representation of *NT* and *AT* a separate LR-ESOM and L-ESOM are used for each class. In this way, the first level has two LR-ESOM: *LRE1-N* and *LRE1-A*, which are trained with *NT* and *AT* separately. In the same way, two L-ESOMs: *LE2-N* and *LE2-A*, are used in the second level. Figure 2 shows the architecture implemented for this problem.

A set of attributes extracted from each TCP/IP packet that composes a connection are the elements of the sequences to feed the architecture. The attributes selected are: TCP flags (*tf*), type of service (*ts*), TCP Windows (*tw*), packet size (*s*), TCP options (*to*) and Time to Live (*tt*), forming the sequence $S1 = \{ \langle tf, ts, tw, s, to, tt \rangle \}$. The following global connection attributes are also included: connection total time (*tc*), total size (*st*), server IP address (*ips*), client IP address (*ipc*), client port (*po*), number of packets in the server (*ps*) and number of packets in the client (*pc*), composing the global vector $G1 = \{ \langle tc, st, ips, ipc, po, ps, pc \rangle \}$.

Training. The data set generated *S1-NT* and *S1-AT* are used to train *LRE1-N* and *LRE1-A* respectively. Once trained, for a sequence in *S1-NT* there would be the compact representations *O1* respect to *LRE1-N* and *O'1* respect to *LRE1-A*. In the same way we get the compact representations *O1* and *O'1* for a sequence in *S1-AT*. The second level *LE2-N* map is trained using as input the extended

² MIT Lincoln Laboratory: http://www.ll.mit.edu/IST/ideval/data/data_index.html.

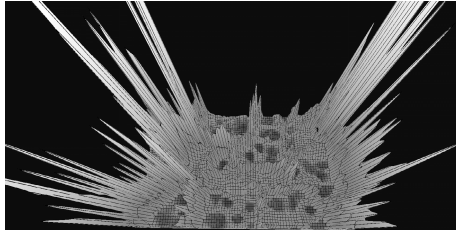


Fig. 3. Visual representation of *LE2-A* using the *L-Matrix* method

Table 1. Detection results obtained compared to other techniques reported in the literature

| Technique | Detection Rate(%) | False Positive Rate(%) |
|---------------|-------------------|------------------------|
| K-NN | 91 | 8 |
| SVM | 98 | 10 |
| SOM hierarchy | 72 | 2 |
| H-ECM | 98 | 0.6 |

vector $E1 = \langle O1, O'1, G1 \rangle$ generated from *S1-NT*, while *LE2-A* is trained with $E1$ generated from *S1-AT*. After training, the maps will give a compact representation $O2$ and $O'2$. Finally, the vector $O = \langle O2, O'2 \rangle$ is used to train the SVM.

Classification. Given a new sequence S and its global attributes vector G , we first compute its extended compact representation at level 1, $E1 = \langle O1, O'1, G \rangle$, where $O1$ comes from *LRE1-N* and $O'1$ from *LRE1-A*. Using $E1$ as input to the second level, $O = \langle O2, O'2 \rangle$ is obtained, which is fed to the SVM to make the classification.

Implementation notes. The H-ECM maps selected have a toroidal grid with 50-by-85 units, $\beta = 0.5$, α is tuned using the entropy changes in the maps, with an initial value of 0.01 and an upper bound of 0.7. The learning parameters γ_1 and γ_2 are taken equal, with an initial value of 0.5 and decreasing linearly to 0.1. The SVM has RBF kernel functions, adjusting the cost parameter ζ and the RBF radius σ with a Grid-search over 10% of the training pattern, randomly chosen. A 15-fold cross validation method for each grid point is performed, resulting in $\zeta = 1.1017$ and $\sigma = 0.0625$. Two thirds of the data are used for training leaving the last third for testing. The H-ECM is implemented in Java 1.5, extending the application Databionics ESOM Tools³ to include recurrent features, the LOF algorithm and the *L-Matrix* 3D visualization. The 3D visual implementation made, used the OpenGL API through the JOGL library.

³ Databionics ESOM Tools: <http://databionics-esom.sourceforge.net>

Results. Table 1 shows the results obtained compare to other techniques used for *Intrusion Detection* [16], [17]. H-ECM equals the best Detection Rate value reported and makes a significant improvement in the False Positive Rate achieved. Figure 3 shows a visual *L-Matrix* representation of the emergent properties extracted with H-ECM. Darker areas represent cluster while peaks show outliers. The tool allows with a mouse click to know the coordinates and LOF value of a point of interest to the user.

4 Conclusions and Future Work

The performance obtained in such a complex problem as the *Intrusion Detection* suggests that H-ECM architecture generates an efficient compression of sequential data without losing either the sequential correlation or long-term dependencies. As stated in [18], the use of compression improves the generalization and prediction properties of the inductive model obtained by the supervised learning algorithm. The *L-Matrix* visualization process also helps in analyzing the properties of the model obtained.

The algorithms are not very demanding in terms of computations. This is mainly due to two factors. First, the architecture does not require a complex preprocessing phase. Secondly, the most costly operations can be done efficiently, e.g. using Search Trees for BMU searching and, the use of values computed during training in the LOF value calculation of a sequence in the classification phase.

The H-ECM architecture is very flexible. Being a hierarchy of functional blocks (ECMs), it may be combined in several ways to match the actual problem structure and needs. It is also possible to use this architecture to process more complex structures than a sequence, such as trees, using approaches similar to the ones presented in [9].

Further research is necessary to understand the emergence phenomena that occur in an ECM, with particular attention to sequential correlation and long-term dependency mechanisms. It is also important to investigate the use of explicit context information in the LOF calculation of the map units after the training phase.

References

1. Dietterich, T.G.: Machine learning for sequential data: A review. In Caelli, T., ed.: Lecture Notes in Computer Science. Volume 2396 of LNCS. Springer-Verlag (2002) 15–30
2. Yaneer, B.: Dynamics of Complex Systems. Addison-Wesley (1997)
3. Kohonen, T.: Self Organizing Maps. Third edn. Springer Verlag (2001)
4. Ultsch, A.: Data mining and knowledge discovery with emergent self-organizing feature maps for multivariate time series (1999)
5. Ultsch, A., Herrmann, L.: The architecture of emergent self-organizing maps to reduce projection errors. In: Proc. of the European Symposium on Artificial Neural Networks. (2005) 1–6

6. Chappell, G.J., Taylor, J.G.: The temporal kohonen map. *Neural Networks* **6**(3) (1993) 441–445
7. Varsta, M., Millan, J., Heikkonen, J.: A recurrent self-organizing map for temporal sequence processing. In: *ICANN '97: Proceedings of the 7th International Conference on Artificial Neural Networks*, London, UK, Springer-Verlag (1997) 421–426
8. Voegtlin, T.: Recursive self-organizing maps. *Neural Networks* **15**(8-9) (2002) 979–991
9. Hagenbuchner, M., Sperduti, A., Tsoi, A.: A self-organizing map for adaptive processing of structured data. *IEEE Transactions on Neural Networks* **14**(3) (2003) 491–505
10. Strickert, M., Hammer, B.: Merge SOM for temporal data. *Neurocomputing* **64** (2005) 39–72
11. Tino, P., Dorffner, G.: Predicting the future of discrete sequences from fractal representations of the past. *Machine Learning* (2001)
12. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. *SIGMOD Rec.* **29**(2) (2000) 93–104
13. Ultsch, A.: Self-organizing neural networks perform different from statistical k-means clustering. In: *Gesellschaft fur Klassifikation.* (1995)
14. Salomon, D.: *Curves and Surfaces for Computer Graphics.* Springer (2005)
15. Vapnik, V.: *The Nature of Statistical Learning Theory.* Springer (2000) Second Edition.
16. Eskin, E., Arnold, A., Prerau, M., Portnoy, L., Stolfo, S.A.: Geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. In Barbara, D., Jajodia, S., eds.: *Applications of Data Mining in Computer Security.* Kluwer (2002)
17. Carrascal, A., Couchet, J., Ferreira, E., Manrique, D.: Anomaly detection using prior knowledge: application to TCP/IP traffic. In Bramer, M., ed.: *IFIP Int'l. Federation for Information Processing. Volume 217., IFIP,* Springer (2006) 139–148
18. Li, M., Vitanyi, P.: *An introduction to Kolmogorov Complexity and its applications.* Second edition edn. Springer (1997)

Locally Scaled Density Based Clustering

Ergun Biçici and Deniz Yuret

Koç University
Rumelifeneri Yolu 34450
Sariyer Istanbul, Turkey
{ebicici, dyuret}@ku.edu.tr

Abstract. Density based clustering methods allow the identification of arbitrary, not necessarily convex regions of data points that are densely populated. The number of clusters does not need to be specified beforehand; a cluster is defined to be a connected region that exceeds a given density threshold. This paper introduces the notion of local scaling in density based clustering, which determines the density threshold based on the local statistics of the data. The local maxima of density are discovered using a k -nearest-neighbor density estimation and used as centers of potential clusters. Each cluster is grown until the density falls below a pre-specified ratio of the center point's density. The resulting clustering technique is able to identify clusters of arbitrary shape on noisy backgrounds that contain significant density gradients. The focus of this paper is to automate the process of clustering by making use of the local density information for arbitrarily sized, shaped, located, and numbered clusters. The performance of the new algorithm is promising as it is demonstrated on a number of synthetic datasets and images for a wide range of its parameters.

1 Introduction

Clustering is the process of allocating points in a given dataset into disjoint and meaningful clusters. Density based clustering methods allow the identification of arbitrary, not necessarily convex regions of data points that are densely populated. Density based clustering does not need the number of clusters beforehand but relies on a density-based notion of clusters such that for each point of a cluster the neighborhood of a given radius (ε) has to contain at least a minimum number of points (φ). However, finding the correct parameters for standard density based clustering [1] is more of an art than science.

This paper introduces the locally scaled density based clustering (LSDBC) algorithm, which clusters points by connecting dense regions of space until the density falls below a threshold determined by the center of the cluster. LSDBC takes two input parameters: k , the order of nearest neighbor to consider for each data point for density calculation and α , which determines the boundary of the current cluster expansion based on its density. The algorithm is robust to background noise and density gradients for a wide range of its parameters.

Density based clustering in its original form, DBSCAN [2], is sensitive to minor changes in its parameters known as the neighborhood of a given radius (ε) and the minimum number of points that need to be contained within the neighborhood (φ). We discuss density based clustering and identify some of its drawbacks in Sect. 2. Although

using different parameters for the radius of the neighborhood and the number of points contained in it appear to give some flexibility, these two parameters are actually dependent on each other. Instead, the LSDBC technique employs the idea of local scaling. We order points according to their distance to their k th neighbor. This gives an approximate measure of how dense the region around each point is. Then, starting with higher density points, we cluster densely populated regions together. The resulting clustering technique does not require fine tuning of parameters and is more robust. OPTICS [2] also bases its clustering decisions on the local density by using kNN type density estimation (differences are explored in Sect. 6).

The local scaling technique, previously employed successfully by spectral clustering [3], makes use of the local statistics of points to separate the clusters within the dataset. The idea is to scale each point in the dataset with a factor proportional to its distance to its k th neighbor. Section 3 discusses local scaling and how it can be used for clustering purposes. We show that when local scaling is used in density based clustering, it creates more robust clusters and allows the automatic creation of clusters without any need for parameters other than k , the order of nearest neighbor to consider, and α , which decides when the drop in the density is necessary for the cluster change.

Density based clustering is important for knowledge discovery in databases. Its practical application areas include biomedical image segmentation [4], molecular biology and geospatial data clustering [5], and earth science tasks [1].

The following lists the contributions of this paper. We introduce locally scaled density based clustering (Sect. 4), which correctly ignores background clutter and identifies clusters within background noise. LSDBC is also robust to changes in the parameters and produces stable clusters for a wide range of them. LSDBC makes the underlying structure of high-dimensional data accessible. The problems we deal with include: (1) finding appropriate parameter values, (2) handling data with different local statistics, (3) clustering in the presence of background clutter, and (4) reducing the number of parameters used. Our results show better performance than prominent clustering techniques such as DBSCAN, k -means, and spectral clustering with local scaling on synthetic datasets (Sect. 5). Our results on image segmentation tasks also show that LSDBC is able to handle image data and segment it into meaningful regions. Related work and density estimation are discussed in Sect. 6 and the last section concludes.

2 Density Based Clustering

Density based clustering differentiates regions which have higher density than its neighborhood and does not need the number of clusters as an input parameter. Regarding a termination condition, two parameters indicate when the expansion of clusters should be terminated: given the radius of the volume of data points to look for, ε , a minimum number of points for the density calculations, φ , has to be exceeded.

Let $d(p, q)$ give the distance between two points p and q ; we give the basic terminology of density based clustering below. ε neighborhood of a point p is denoted by $N_\varepsilon(p)$ and is defined by $N_\varepsilon(p) = \{q \in Points \mid d(p, q) \leq \varepsilon\}$, where *Points* is the set of points in our dataset. A *core point* is defined as a point above the density threshold wrt. ε and φ , i.e. $|N_\varepsilon(p)| \geq \varphi$. A *border point* is defined as a point below the threshold but that belongs to the ε neighborhood of a core point.

Definition 1 (Directly density-reachable). A point p is directly density reachable from a point q wrt. ϵ and φ , if $p \in N_\epsilon(q)$ and $|N_\epsilon(q)| \geq \varphi$ (core point condition).

Definition 2 (Density-reachable)

A point p is density reachable from a point q wrt. ϵ and φ , if there is a chain of points $p_1, p_2, \dots, p_n, p_1 = q, p_n = p$ such that p_{i+1} is directly density reachable from p_i .

Definition 3 (Density-connected)

A point p is density connected to a point q wrt. ϵ and φ , if there is a point r such that both p and q are density reachable from r wrt. ϵ and φ .

A cluster C wrt. ϵ and φ is a non-empty set of points such that $\forall p, q \in C, p$ is density connected to q wrt. ϵ and φ .

Selecting appropriate parameters, ϵ and φ , is difficult in DBSCAN and even in the best setting, the results may not be good. Figure 1 gives representative results using our synthetic datasets. Note that minor changes in the parameters ϵ and φ creates spurious clustering results. In all of the following graphics, gray points are considered as noise.

Ester *et al.* [11] suggest that the user will look at the sorted 4-dist graph (plot of points' distance to their 4th nearest neighbor in descending order) and select a threshold point, which will divide the points into two sets: noise and clusters. The selected threshold, $4NNDist$ value, can be used for determining the parameters as in: $\epsilon = 4NNDist$ and $\varphi = 4$. However, for some datasets, the threshold point may not be easy to pick, it may not be unique if there is variance in the density, $k = 4$ may not be the ideal setting, and this approach assumes user intervention.

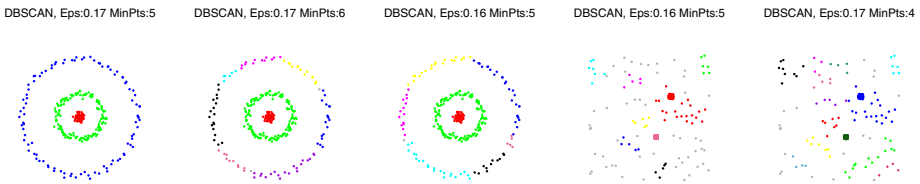


Fig. 1. Density based clustering is sensitive to minor changes in ϵ and φ

3 Local Scaling

Zelnik-Manor and Perona [3] successfully applied local scaling to spectral clustering. Local scaling is a technique which makes use of the local statistics of the data when identifying clusters. This is done by scaling the distances around each point in the dataset with a factor proportional to its distance to its k th nearest neighbor. As a result, local scaling finds the scale factors for clusters with different densities and creates an affinity matrix in which the affinities are high within clusters and low across clusters.

Given two points x_i and x_j from a dataset, X , let A_{x_i, x_j} denote the affinity between the two points, showing how similar two objects are. Based on [6], $\forall x_i, x_j \in X$, let the following properties hold:

$$A_{x_i, x_j} \in [0, 1], A_{x_i, x_i} = 1, A_{x_i, x_j} = A_{x_j, x_i}. \tag{1}$$

We could define A_{x_i, x_j} as:

$$A_{x_i, x_j} = \exp\left(-\frac{d^2(x_i, x_j)}{\sigma^2}\right), \quad (2)$$

where $d(x_i, x_j)$ is any distance function (such as the Euclidean ($\|x_i - x_j\|^2$) or the cosine between feature vectors) and σ is a threshold distance below which two points are thought to be similar and above which two points are considered dissimilar. A single scaling parameter, σ , may not work for the whole dataset when clusters with different densities are present. Instead, a local scaling parameter σ_i can be calculated for each data point x_i such that the affinity between a pair of points, x_i and x_j , is given by:

$$\hat{A}_{ij} = \exp\left(-\frac{d^2(x_i, x_j)}{\sigma_i \sigma_j}\right), \quad (3)$$

where $d(x_i, x_j)$ corresponds to the distance from x_i to x_j . When selecting the local scaling parameter σ_i , local statistics of the neighborhood of point x_i is considered. The choice in [3] is:

$$\sigma_i = d(x_i, x_i^k), \quad (4)$$

where x_i^k is the k th closest neighbor of point x_i and k is chosen to be 7. Thus, $\sigma_i = 7NNDist(x_i)$ in spectral clustering with local scaling.

4 Locally Scaled Density Based Clustering

Locally scaled density based clustering algorithm clusters points by connecting dense regions of space until the density falls below a threshold determined by the center of the cluster. LSDBC takes two input parameters, k , the order of nearest neighbor to consider for each point in the dataset for density calculation and α , which determines the boundary of the current cluster expansion based on its density.

The LSDBC algorithm first calculates the ε values for each point based on their kNN distances. ε allows us to order points based on their density. Smaller ε values correspond to denser regions in the dataset. The set of points are then sorted in ascending order of their ε . Algorithm 1 presents the main method of LSDBC. The function $kNNDistVal$ takes a point and a number k and returns the distance of the point to its k th nearest neighbor, ε , as well as the set of its k nearest neighbors. $localMax$ function ensures that the selected point is the most dense point locally in its neighborhood.

The $ExpandCluster$ procedure, given in Algorithm 2, expands the cluster of a given point, p , by exploring neighboring points and placing them into the same cluster as p when their density is above $\frac{density(p)}{2^\alpha}$. The initial point p is called the center point of the cluster. The α parameter prevents the expansion of a given cluster into regions of points with a density smaller than a factor of $2^{-\alpha}$ relative to the center. The density of a given point p is calculated as:

$$density(p) = \frac{k}{\varepsilon^n}, \quad (5)$$

Input: D : Distance matrix, k : input to kNN -dist function, n : number of dimensions, α .

Output: Allocation of points to clusters.

```

for  $p \in Points$  do
   $p.class = UNCLASSIFIED$ ;
   $[p.Eps, p.neighbors] = kNNDistVal(D, p, k)$ ;
end
 $Points.sort()$ ; /* Sort on Eps */
 $ClusterID = 1$ ;
for  $p \in Points$  do
  if  $p.class == UNCLASSIFIED$  and  $localMax(p)$  then
     $ExpandCluster(p, ClusterID, n, \alpha)$ ;
     $ClusterID = ClusterID + 1$ ;
  end
end

```

Algorithm 1. LSDBC: Locally Scaled Density Based Clustering Algorithm

where n corresponds to the dimensionality of the dataset. A point p' is defined as a *core point* if its density exceeds the density of the center point for the cluster multiplied by $2^{-\alpha}$:

$$\frac{k}{2^{\alpha} \varepsilon_p^n} \leq \frac{k}{\varepsilon_{p'}^n}. \quad (6)$$

Therefore,

$$\varepsilon_{p'} \leq 2^{\alpha/n} \varepsilon_p. \quad (7)$$

Equation (7) provides us a cutoff point when expanding cluster regions.

In the final clustering scheme of LSDBC, we need only two parameters: the k value, which is a parameter corresponding to up to which closest neighbor we should look for when clustering points, and α which takes role in identifying a cutoff density for the cluster expansion. The focus of this paper is to automate the process of clustering by making use of the local density information for arbitrarily sized, shaped, located, and numbered clusters. LSDBC enjoys good clustering results for a wide range of values for k and α . An obvious advantage of LSDBC is that it is not sensitive to background density variations and therefore, it can be used with a wide range of clustering problems.

Computational Complexity. $kNNDistVal$ operates in $O(n)$ time. As it was suggested in [11], if we use a tree-based spatial index, k nearest neighbors can be retrieved in $O(\log n)$ time. Therefore, the run-time complexity of LSDBC algorithm is $O(n \log n)$, which is the same as DBSCAN's run-time complexity.

5 Experiments

We compared our algorithm, LSDBC with (1) the original density based clustering algorithm, DBSCAN, (2) spectral clustering with local scaling, and (3) k -means clustering. The results show that LSDBC's performance is superior to these three clustering techniques on the problems we analyzed. Spectral clustering with local scaling achieves comparable performance on some of the synthetic datasets. LSDBC produces robust

```

Input: point, ClusterID, n: number of dimensions,  $\alpha$ .
point.class = ClusterID;
Seeds = point.neighbors;
for currentP  $\in$  Seeds do
    if currentP.class == UNCLASSIFIED then
        currentP.class = ClusterID;
    else
        Seeds.delete(currentP);
    end
end
while Seeds.length > 0 do
    currentP = Seeds.first();
    if currentP.Eps  $\leq 2^{\alpha/n} \times$  point.Eps then
        Neighbors = currentP.neighbors;
        for neighborP  $\in$  Neighbors do
            if neighborP.class == UNCLASSIFIED then
                Seeds.append(neighborP);
                neighborP.class = ClusterID;
            end
        end
    end
    Seeds.delete(currentP);
end

```

Algorithm 2. *ExpandCluster*: Expands the cluster of a given point

clusters for a broad range of values for k and α . The robustness of LSDBC for different values of k can be seen in Fig. 2.

In this set of experiments, we compare the results of LSDBC with the clustering techniques of DBSCAN, k -means, and spectral clustering with local scaling. For each dataset, k -means and spectral clustering methods accept the number of clusters as input where $k = 20$ is the number of ideal clusters for the given dataset. Therefore, for the clustering methods that we compare, including DBSCAN, we chose the best possible setting for the clustering. In Fig. 3 we compare their performance on a more complex dataset. In all of these examples, the performance of LSDBC is superior to others in terms of its ability to respect the boundaries of closely located and similarly populated regions. The difference between LSDBC's results is that when $k = 6$ or $k = 7$, the background cluster is divided into 3 and 2 clusters respectively whereas when $k = 8$, they form a single cluster. When we look at the results of k -means and spectral clustering with local scaling, even under the best settings, we can see that they divide densely populated regions into separate clusters and merge regions with different densities together whereas DBSCAN classifies regions with lower density below a threshold as noise.

Apart from generating unnatural clusters in some datasets, another drawback of spectral clustering and k -means is their requirement of the number of clusters as input, whereas LSDBC can discover clusters of arbitrary size, shape, location, and number without any knowledge of the number of clusters in the data.

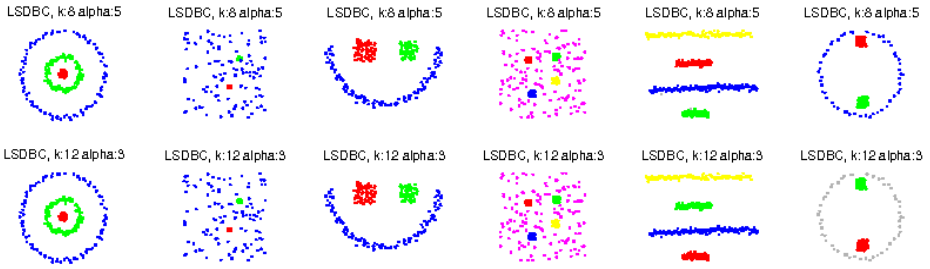


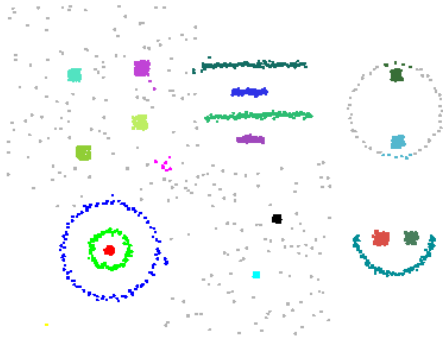
Fig. 2. Robustness of LSDBC for different values of k and α

In our next set of experiments, we deal with the task of image segmentation. The results can be seen in Figs. 4, 5, 6, and 7. LSDBC is able to decipher transparency in the images. For instance, the background seen through the handle hole in the still life image (Fig. 4) and the background itself is clustered into the same cluster. The resulting clusterings can be used to summarize images as well as compress them. As can be seen from the resulting clusterings, LSDBC provides an adequate separation of the original images, which makes it useful for the task of filtering trees [7] (see Fig. 5).

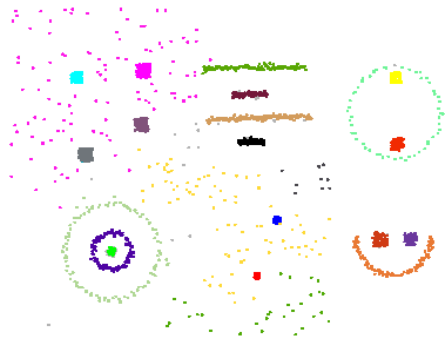
6 Related Work and Density Estimation

In Ester *et al.* [1], density based clustering (DBSCAN) was presented as a clustering technique which can discover clusters of arbitrary shape. Hinneburg and Keim [8] introduced a new density based clustering technique as DENCLUE, which sums the density impact of a data point within its neighborhood. In effect, density based clustering methods estimate the density of points in a given dataset to cluster and differentiate densely populated regions. Two methods are commonly used for *density estimation*: Parzen windows and k -nearest neighbor (kNN) estimation [9]. In Parzen windows, we assume the existence of a d dimensional hypercube with volume $V = \varepsilon^d$, where ε is the length of an edge of the hypercube. Then, the number of points falling within this volume gives an estimate of the density (pick a radius and count the number of neighbors). Problems arise when choosing ε , which determines the volume, V , also known as the problem of finding the best window size. In kNN , we choose k nearest neighbors and grow the corresponding ε and volume V until it encloses the $k + 1$ points (pick a number of neighbors and compute the radius).

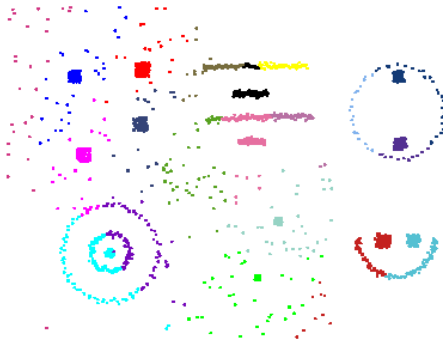
Density based clustering algorithms can also be divided into two types based on how they estimate the density: Parzen window type and kNN type. Among the Parzen window type approaches, we can count DBSCAN [1], DENCLUE [8], and CLIQUE [10]. Most of these algorithms suffer from the problem of choosing the best window size. LSDBC and OPTICS [2] are both kNN type density based clustering algorithms. OPTICS focuses on providing a representation of data (e.g. reachability plots) that enables different clusterings to be implemented and generates an ordering of points based on the order in which they are explored by the subroutine ExpandClusterOrder. LSDBC starts with a density based ordering and performs cluster expansions starting with the



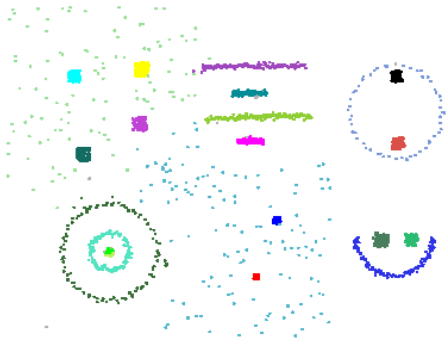
DBSCAN, $r=0.04$, $\text{density}=5$:
Only highly dense regions are recognized.



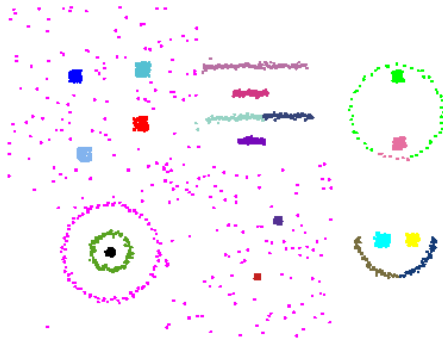
LSDBC, $k=6$, $\alpha=3$:
Background clutter is divided into 3 clusters.



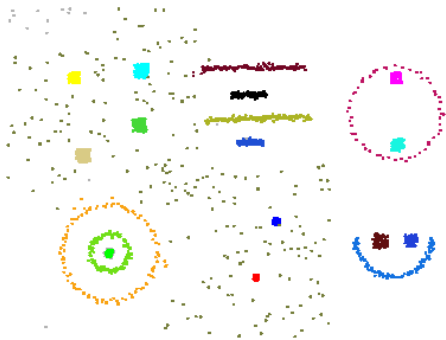
k-means, $k=20$:
Densely populated regions are divided.



LSDBC, $k=7$, $\alpha=3$:
Background clutter is divided into 2 clusters.



Spectral with local scaling, $k=20$:
Densely populated regions are divided,
diversely populated regions merged.



LSDBC, $k=8$, $\alpha=3$:
Background clutter is classified as a single
cluster.

Fig. 3. Comparison of clustering performance on a dataset with different local statistics

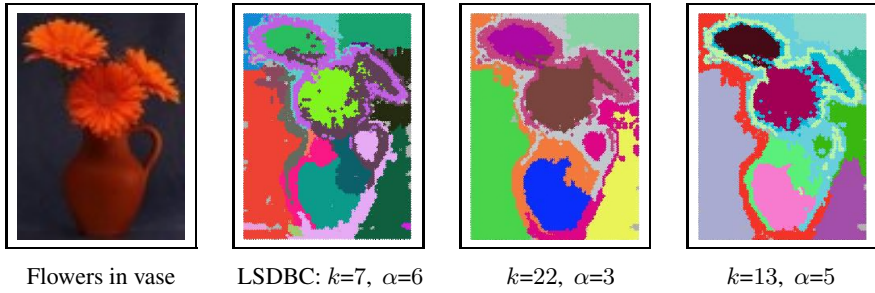


Fig. 4. Segmentation of a still life image. Notice the identification of the same transparent regions, which can be seen through the handle of the vase

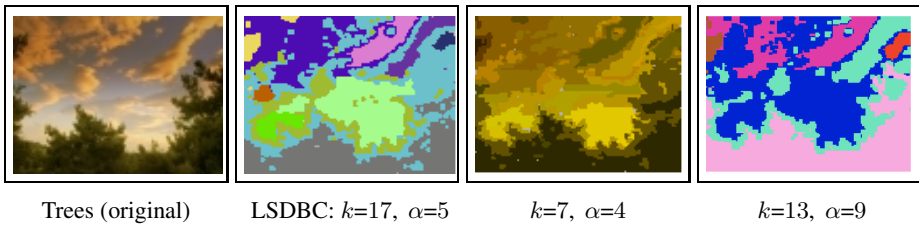


Fig. 5. Segmentation of a group of trees and the sky

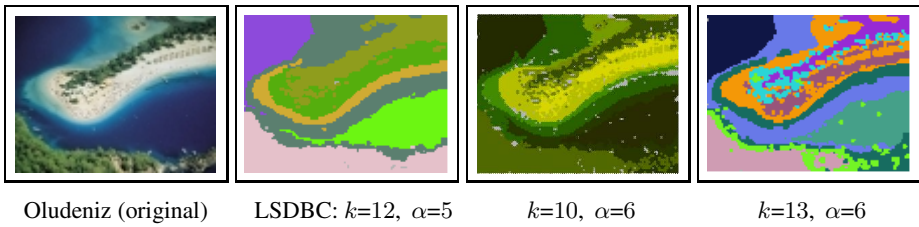


Fig. 6. Segmentation of an image of a seaside, Oludeniz

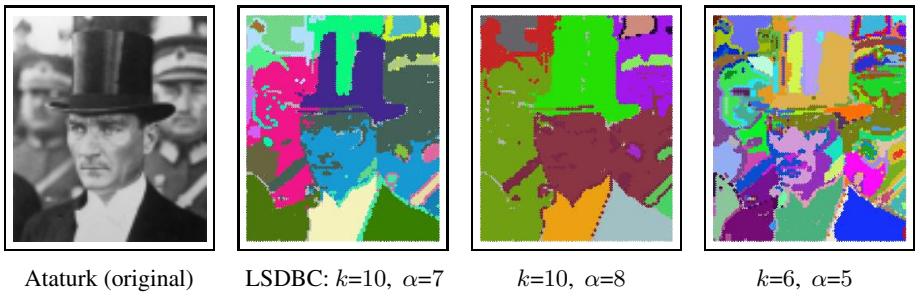


Fig. 7. Segmentation of an image of Ataturk

densest available point. Also, the cut-off for clusters in OPTICS is decided based on the density gradient of the edges of clusters, whereas LSDBC bases its cut-off on the density of the center of the cluster, which we believe to be more robust and noise free.

7 Conclusion

We have introduced the locally scaled density based clustering method. LSDBC discovers local maxima of density using a k-nearest-neighbor density estimation method and grows each cluster until the density falls below a pre-specified ratio of the center point's density. The resulting clustering technique is able to identify clusters of arbitrary shape on noisy backgrounds that contain significant density gradients. The performance of the new algorithm is demonstrated on a number of synthetic datasets and real images and shown to be promising for a broad range of its parameters.

LSDBC can be effectively used as a tool for summarizing the inherent relationships within the data. We have shown that LSDBC's performance in differentiating between densely populated regions is better than other clustering algorithms that we considered. LSDBC can also be used to summarize and segment images into meaningful regions.

References

1. Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231, 1996.
2. Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: ordering points to identify the clustering structure. In *SIGMOD '99: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, pages 49–60, New York, NY, USA, 1999. ACM Press.
3. Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Eighteenth Annual Conference on Neural Information Processing Systems*, 2004.
4. M. Emre Celebi, Y. Alp Aslandogan, and Paul R. Bergstresser. Mining biomedical images with density-based clustering. In *ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing*, volume I, pages 163–168, Washington, DC, USA, 2005. IEEE Computer Society.
5. Jörg Sander, Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998.
6. Pietro Perona and William T. Freeman. A factorization approach to grouping. In *ECCV '98: Proceedings of the 5th European Conference on Computer Vision*, volume I, pages 655–670, London, UK, 1998. Springer-Verlag.
7. Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. *SIGMOD Record*, 25(2):103–114, 1996.
8. Alexander Hinneburg and Daniel A. Keim. An efficient approach to clustering in large multimedia databases with noise. In *KDD*, pages 58–65, 1998.
9. Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.
10. Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *SIGMOD '98: Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA*, pages 94–105. ACM Press, 1998.

Hierarchical Rules for a Hierarchical Classifier^{*}

Igor T. Podolak

Institute of Computer Science, Faculty of Mathematics and Computer Science,
Jagiellonian University
Nawojki 11, Kraków, Poland
uipodola@theta.uoks.uj.edu.pl

Abstract. A system of rule extraction out of a complex hierarchical classifier is proposed in this paper. There are several methods for rule extraction out of trained artificial neural networks (ANN's), but these methods do not scale well, *i.e.* results are satisfactory for small problems. For complicated problems hundreds of rules are produced, which are hard to govern.

In this paper a hierarchical classifier with a tree-like structure and simple ANN's at nodes, is presented, which splits the original problem into several sub-problems that overlap. Node classifiers are all *weak* (*i.e.* with accuracy only better than random), and errors are corrected at lower levels. Single sub-problems constitute of examples that were hard to separate. Such architecture is able to classify better than single network models.

At the same time IF-THEN rules are extracted, which only answer which sub-problem a given example belongs to. Such rules, by introducing hierarchy, are simpler and easier to modify by hand, giving also a better insight into the original classifier behaviour.

1 Introduction

A *classifier* in machine learning is a model built to reflect the implicit rules hidden in a training file [1,2]. A classifier has to assign correct classes (from a finite set of them) to examples previously unseen, *i.e.* such that were not used during the training phase. Several methods are used, one of them being artificial neural networks. ANN's show a high rate of generalisation, *i.e.* the ability to correctly classify previously unseen examples, but at the same time give no clues as to the possible structure of the problem, and do not answer why a given class was selected, *i.e.* have low *low explanatory* power [2,3]. At the same time, ANN's usually require a lot of training just to find the correct network architecture.

This paper shows how a hierarchical classifier (HC) can be built automatically (its' scheme was proposed earlier by the author in [4]), and how a set of IF-THEN rules can be extracted that would reflect the classifier behaviour. The main objectives are, first, to show that an HC can be easily built out of weak classifiers, then that rules can be extracted that would reflect the problem structure. This

^{*} Research was funded by Jagiellonian University's grant "Hierarchical classifiers".

approach puts the classifier in the line of other committee machines, such as the Hierarchical Mixture of Experts or AdaBoost models [2,5].

The paper is divided as follows: first a definition of the hierarchical classifier is given, then its construction is given in more detail, after that the task of rule extraction is defined followed by the description of the algorithm used, and the structure of the rules. Experiments and discussion follow.

2 Definition of the Problem and Model

Definition 1. For a given training data example pairs from set $\mathcal{X} = \{(x_k, c_k)\}$, where $x_k \in \mathbb{R}^p$ is an input vector, and $c_k \in C$, from a finite set $C = \{C^i\}_{i=1}^K$, is a nominal value, then a classifier Cl is a mapping

$$Cl : \mathcal{X} \mapsto C \quad (1)$$

One of the main methods to build such a model is to use the feed-forward network paradigm. Main problem is that one cannot say which architecture will fit the problem at hand best. There are no clear clues as to how many neurons should be used, therefore the network designer usually has to use either a network pruning technique, or try several networks. Both approaches are costly.

In the presented solution, the classifier has a tree-like structure, with weak networks at each node. Root node classifier node is trained to solve the whole problem, but because of its weak nature, the responses found are only a little (but significantly) better than of a classifier, which assigns the most frequent class to all examples [6,2]. It needs to be observed that *the incorrect answers are not random*. The hypothesis is that if C_i is returned as an incorrect answer, it probably is a class, that is similar to the true class, a class that the classifier finds most similar. Therefore groups of such similar classes constitute natural sub-problems.

After introductory training, clusters of classes that are frequently mistaken with each other are identified, and a new sub-problem is formed out of each one. Then a new ANN node is trained for each of the sub-problems. In the presented model the clusters *overlap*, which helps to achieve a better overall accuracy.

2.1 Detailed Description

The proposed model has a tree-like structure with classifiers $Cl^i : X^i \mapsto C^i$ at nodes, which map examples from $X^i \subset X$ into a subset of classes $C^i \subset C$ (for the root $C \equiv C$). Classes are combined into clusters of classes that are frequently mistaken, *e.g.* classes C_a and C_b are in the same cluster, if Cl^i frequently gives C_a as the result when C_b is the true class, and the opposite. In such case it is said that classes C_a and C_b are *Cl-similar*. This grouping is accomplished through simple clustering of the C^i 's classifier confusion matrix, modified for the case of overlapping clusters. In other words, a set of m clusters

$$\mathcal{Q}^i = \{Q_j^i\} \quad j = 1, \dots, m \quad (2)$$

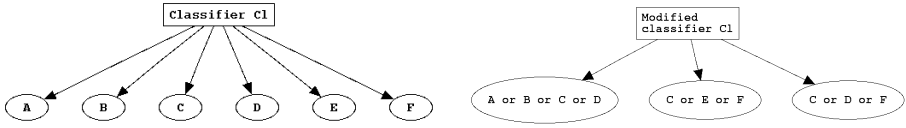


Fig. 1. An example original classifier Cl and a modified one Cl_{mod}

is found, where a single cluster Q_j^i consists of classes frequently mistaken

$$Q_j^i = \{C_l^i \in C^i | l = 1, \dots, n_{Q_j^i}\} \tag{3}$$

where $n_{Q_j^i}$ is the number of classes in cluster Q_j^i .

The already trained classifier Cl^i is modified into classifier $Cl_{mod}^i : \mathcal{X} \mapsto Q^i$, *i.e.* one that returns the cluster to which the true class belongs to (see Fig. 1). Individual node classifiers are weak and have low accuracy, but there is a high chance that the correct cluster of classes is found, therefore narrowing down the possible answers. A notion of generalised accuracy is introduced

Definition 2. *The generalised accuracy is the rate of examples which are assigned by classifier Cl into a cluster including examples' true classes.*

Individual classifiers may have low standard accuracy, *i.e.* the ratio of correctly identified examples, but high generalised accuracy. *E.g.* in one model built for the UCI vowel problem [7], the root classifier, with only 2 hidden nodes achieves only a 38.38% standard accuracy rate, but at the same time a 99.90% generalised accuracy. This shows that the heuristic about similarity of true and incorrect classes is true. The whole 3 level hierarchical classifier achieves a 96.26% accuracy!

The cluster construction is equivalent to problem partitioning into overlapping sub-problems. Only these examples that have true classes from the given cluster are used, therefore the problem gets smaller at each level. Addition of consecutive layers continues up to the moment when the accuracy of solution is satisfactory. Therefore, the proposed model frees the classifier developer from a “trial-and-error” approach where several of different architectures need to be checked.

2.2 Clusters and Combination of Individual Results

Each cluster has less than half of the parent node classes. The clusters are found by inspecting the classifier’s confusion matrix and clustering classes that are frequently mistaken.

The clusters are found so that they *overlap*, that is a class $C_l^i \in C^i$ may belong to more than one cluster, or there may exist k and l , $k \neq l$, such that the intersection $Q_k^i \cap Q_l^i \neq \emptyset$. If the clusters were crisp, then if a wrong was selected (*i.e.* one that does not include the true class), then there would be no chance to correct the initial answer at a lower level. Since the clusters overlap, the chances

are much higher. This is very important if individual node networks are very simple, *e.g.* have a very low number of hidden neurons, since such classifiers frequently do not return as output all of the classes. For example, in the above mentioned model built for the vowel problem, the root classifier recognised only 4 classes out of all 11 (usually these that are the most frequent). This is important when the classifiers are weak – at first steps of learning neural networks return only the most frequent classes, especially if the network resources, *e.g.* number of hidden neurons, are scarce. A new child classifier is built for each cluster, until a maximum level is reached, there are less than three classes in the cluster, or the accuracy is satisfactory.

The hierarchical classifier works in 2 phases: first the calculation of consecutive layers, then a step of combining bottom-level results. If classifier Cl^i is an ANN, then its normalised output can be regarded as individual classes posterior probabilities $P(Cl^i|x)$, where x is the input vector of features. If Cl^i recognises n_{Q^i} clusters Q^i_j , then for each cluster a sum

$$PQ(Q^i_j|x) = \sum_{C_l^i \in Q^i_j} P(C_l^i|x) \tag{4}$$

can be computed, and a renormalised vector $P(Q^i|x) = [P(Q^i_1|x), \dots, P(Q^i_{n_{Q^i}}|x)]$ can be treated as individual clusters posterior probabilities given input x .

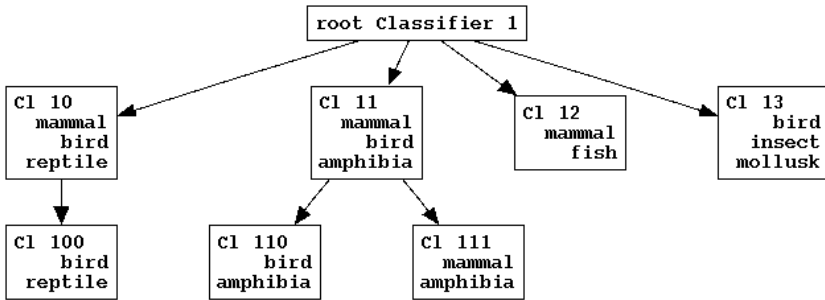


Fig. 2. The complete hierarchical classifier architecture for the zoo problem

This process is repeated at each tree, and after the “forward” run individual class and cluster probabilities at each node are found. After reaching the leaf level, the results are “back-propagated” recursively using the following formula (for a two-level classifier, for simplicity, and root cluster $Q \equiv \bigcup_j Q^j$)

$$P(C_l|x) = \sum_{i=1}^{n_Q} P(Q^i|x)P_{Q^i}(C_l^i|x) \tag{5}$$

where n_Q is the number of clusters of the root classifier, $P(Q^i|x)$ is the probability of cluster Q^i , and $P_{Q^i}(C_l^i|x)$ are the individual posterior class probabilities in

cluster Q^i found by classifier Cl^i . This methodology is similar to other committee machines [8,9]. An example classifier architecture is shown in Fig. 2.

3 Hierarchical System of Rules

ANNs achieve high accuracy rates, but frequently have low generalisation rate in cases when number of parameters exceeds the number of training examples, also have low self-explanatory power. It is also hard to include some prior knowledge about the problem into the network design [10,11,12]. Therefore extraction of rules may come to rescue.

Several methods are used for extraction of IF-THEN rules out of trained ANN's. In case of large training sets and hard problems with several output classes, the extracted rules are not easily readable. The algorithms which give clear rules for simple problems such as *iris* [7], but for a more complicated problem give a large set of complicated rules. *I.e.* the extraction methods do not scale well with the growing number of input attributes and output classes. The proposed hierarchical architecture, together with a rule extraction system is fit for such tasks.

The system proposed here utilises a modified Setiono's FERNN algorithm [3,4,12] (in the version used here the network training algorithm was changed to Resilient Propagation). A FERNN algorithm trains ANNs pruning to arrive at a simple network with low number of connections, therefore low number of rules. The trained ANN is then examined to generate rules of the general form

$$\text{IF } Pred(x) \text{ THEN } Class(x) = C_i \tag{6}$$

where $Pred(x)$ is some logical predicate over the input feature vector x .

In the proposed architecture two kinds of rules are extracted out of each node classifier Cl^i : first rules which give that network's prediction of the final class

$$\text{IF } Recognised(Q^i) \wedge PredClass_k^{Cl^i}(x) \text{ THEN } Class(x) = C_l \in \bigcup_j Q_j^i : Q_j^i \in Q^i \tag{7}$$

where $Recognised(Q^i)$ is a predicate stating that the output class is one of classes included in $\bigcup_j Q_j^i$, $PredClass_k^{Cl^i}(x)$ is some class predicate, and $C_l \in Q^i$ is some class recognised by classifier $Cl^i \in \bigcup_j Q_j^i$, then other rules which predict clusters of classes

$$\text{IF } Recognised(Q^i) \wedge PredCluster_m^{Cl^i}(x) \text{ THEN } Cluster(x) = Q_n^i \in Q^i \tag{8}$$

where $PredCluster_m^{Cl^i}(x)$ is some classifier recognising a cluster. The value of $PredCluster_m^{Cl^i}(x)$ is computed by a parent level classifier. Both predicates $PredClass_k^{Cl^i}(x)$ and $PredCluster_m^{Cl^i}(x)$ are computed *only* if the final class is probably in cluster Q^i .

For the *zoo* [7] problem (animal classification), apart from the rules finding final classes, a set of cluster rules is obtained, *e.g.*

```

if(+3.03*"eggs" -2.14*"milk" -1.0*"legs" +3.84*"tail" <= 1.7) {
  if(-2.23*"feathers" +3.56*"eggs" -2.26*"milk" +2.55*"aquatic" -1.0*"catsize" <= 2.88)
  {
    cluster = "mammal OR sea animal"
  } else {
    if( ... another expression ...) {
      cluster = "fish OR insect"
    } else {
      cluster = "reptile OR mollusk"
    }
  }
} else {
  . . .
}

```

There are much less rules for clusters, then there are for individual classes. After minimisation of the logical expressions, there are about half as many cluster rules, as there are simple class rules. The cluster rules include much more general information which may be corrected by hand to include some *prior-knowledge* on the given problem, information which is complicated and hardly readable when simple neural networks are used.

4 Experiments

A number of experiments were performed to compare the accuracy of the original hierarchical classifier, and the accuracy of the derived set of IF-THEN rules. To accomplish that, a number of classifiers were trained using training sets from the UCI machine learning repository [7,13], all of which have a high number of output classes, therefore especially fit for the proposed classifier architecture. All the classifiers were set to use at most 3 levels of classifiers (including the root classifier level). To show that classifiers can be built even with very small individual networks, only 1 to 5 hidden feed-forward networks were used.

To extract rules, stratified example sets were constructed for each network, then hidden neurons' activations recorded, and decision trees built for such training sets, separately for class and cluster predictions (an *oracle* approach [11]). Because of such approach, the extracted rules reflect the neural network's actual performance (previously only activations for correct classifications were used [4]). The rules were additionally pruned to minimise their number. To mimic a rule expert system, the rules were rewritten into the Jess language, and checked with the Jess inference engine (a Rete algorithm) [14].

The main objective was to compare the accuracy of extracted system of rules with the original classifier. A comparison of hierarchical classifier, extracted rules, and Setiono's N2CS2 [3] single network with optimised number of neurons accuracies is given in Table 1. It can be seen that that the hierarchical classifier compares very well with single ANN's with optimised number of hidden neurons [4,15]. The proposed classifier makes it possible to quickly build a network architecture, combined of small and weak classifiers, and with equal or superior accuracy. The graphs are shown in figure 3.

It can be seen that even for networks with 1 hidden neuron, the proposed architecture is able to give better than expected results, *e.g.* the one built for

Table 1. Experiment results for some training sets from the UCI Repository [713] compared with results (last column) of Setiono’s N2CS2 optimised algorithm [3]. Means out of 10 runs each are given for 1 to 5 hidden individual ANN’s, and for rule systems extracted. All accuracies are in percentages.

| | 1 hidden | | 2 hidden | | 3 hidden | | 4 hidden | | 5 hidden | | Setiono train |
|---------------|----------|-------|----------|-------|----------|-------|----------|-------|----------|-------|---------------|
| | HC | rules | HC | rules | HC | rules | HC | rules | HC | rules | |
| audiology | 59.3 | 30.9 | 85.6 | 47.2 | 92.6 | 44.2 | 95.0 | 71.5 | 95.7 | 84.4 | 95.5 |
| primary-tumor | 35.1 | 24.8 | 53.7 | 37.5 | 65.6 | 44.9 | 70.0 | 47.2 | 73.3 | 48.8 | 62.1 |
| vowel | 44.4 | 13.7 | 88.8 | 67.7 | 95.3 | 84.6 | 97.0 | 88.1 | 97.9 | 90.5 | 94.3 |
| zoo | 82.8 | 63.0 | 93.4 | 86.8 | 99.4 | 96.9 | 99.7 | 99.5 | 99.7 | 99.7 | 100 |

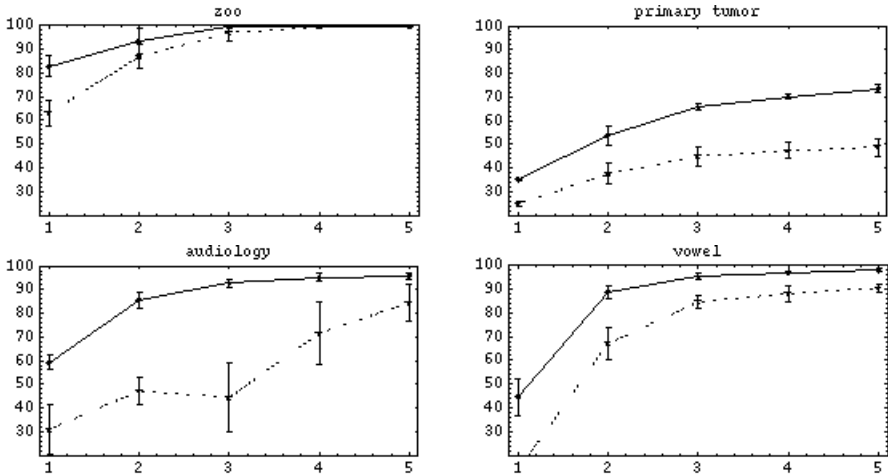


Fig. 3. Accuracy graphs for the hierarchical classifier and derived systems of rules trained for zoo, primary tumor, audiology and vowel problems [713]. The classifiers were built using networks of 1 to 5 hidden neurons. Classifiers were built 10 times per each number of hidden neurons, and standard deviations are shown.

vowel problem (with 11 output classes with equal prior probability) achieves a 44% accuracy over the training set, and even 80% for the zoo problem. This happens even though the root classifiers may have only a base level accuracy, frequently predicting *only* two out of all classes. Thanks to the hierarchical architecture and overlapping clusters, this is enough to separate the problem into several simpler sub-problems, where the small networks are able to discern examples. Classifiers with only few hidden neurons at each node give comparable or superior results. Even for such a complicated task as the *primary-tumor* [713], for which neural networks very rarely achieve a level above 60%, the HC achieved significantly better accuracy of over 70%.

The rule derived rule systems for classifiers with small hidden layers have much lower accuracy (and with high deviation), but for larger single node networks the accuracy is only 10 – 20% lower, even equalling that of base classifier for simpler

problems (*zoo*). One must remember that these are only base rule systems which may be extended by inclusion of prior knowledge. Accuracy for a complicated problem like *primary-tumor* is much lower here since individual node classifiers are very weak. For such problems and rule extraction, the individual classifiers need to have better accuracy – this follows from the fact that with weak classifiers selection of output class is made from a small subset of classes.

The whole software was written in a Weka machine learning environment [16]. Since the individual nodes at each level are independent, the parallelisation of the program was straightforward, speeding up calculations.

5 Discussion

The hierarchical classifier achieves very good results, thanks to the combination of weak classifiers. Clusters constructed reflect the problems each ANN had with the problem given, therefore automatically giving a problem partition. It is also possible to extract a hierarchical system of IF–THEN rules, where the problem and its partition are reflected. A standard expert system may be built, ready for inclusion of some prior knowledge available.

The rule system still has lower accuracy than the original hierarchical classifier. This is easily seen, especially when the number of hidden neurons is very low (*e.g.* only 1 hidden neuron). It still is a problem of the accuracy of rule extraction, as well as the construction of the whole rule system.

There is still much to do: better ANN training algorithm, especially for low number of neurons and its increased cooperation with the clustering algorithm, better construction of the rule system, experiments with larger data sets. It still remains to be proved, that such methodology gives a universal classifier, that is able to give accuracies better than a given ε .

References

1. D. Hand, H. Mannila, P.S.: Principles of Data Mining. MIT Press (2001)
2. T. Hastie, R. Tibshirani, J.F.: The Elements of Statistical Learning. Springer Verlag (2001)
3. Setiono, R.: Feedforward neural network construction using cross validation. *Neural Computation* **13** (2001) 2865–2877
4. I. T. Podolak, S. Biel, M.B.: Hierarchical classifier. In Wyrzykowski, R., ed.: *Parallel Processing and Applied Mathematics*. LNCS, Springer (2006) 591–598
5. Schapire, R.: The strength of weak learnability. *Machine Learning* **5** (1990) 197–227
6. Haykin, S.: *Neural networks, a comprehensive foundation*. Prentice Hall (1999)
7. D. J. Newman, S. Hettich, C.L.B.: *Uci repository of machine learning databases* (1998)
8. J. Kittler, A. Hojjatoleslami, T.W.: Strategies for combining classifiers employing shared and distinct pattern representations. *Pattern Recognition Letters* **18** (1997) 1373–1377
9. E. Bax: Validation of voting committees. *Neural Computation* **10**(4) (1998) 975–986

10. R. Andrews, J. Diederich, A.B.T.: Survey and critique of extracting rules from trained artificial neural networks. *Knowledge-Based Systems* **8**(6) (1995) 373–389
11. W. Duch, R. Setiono, J.M.Z.: Computational intelligence methods for rule-based data understanding. *Proceedings of the IEEE* **92**(5) (2004) 771–805
12. R. Setiono, W.K.L.: FERNN: An algorithm for fast extraction of rules from neural networks. *Applied Intelligence* **12**(1-2) (2000) 15–25
13. M. Zwitter, M.S.: Primary tumor data set (1988)
14. Friedman-Hill, E.: Jess, the rule engine for java platform. Sandia National Laboratory (2001)
15. Podolak, I.T.: Hierarchical classifier with overlapping class groups. *Expert Systems with Applications* **34**(3) (2007)
16. I. H. Witten, E.F.: *Data mining: practical machine learning tools with Java implementations*. Morgan Kaufmann, San Francisco (2000)

A Demonstration of Clustering in Protein Contact Maps for Alpha Helix Pairs^{*}

Robert Fraser¹ and Janice Glasgow²

¹ University of Waterloo, Waterloo, ON, Canada, N2L 3G1

r3fraser@cs.uwaterloo.ca

<http://www.cs.uwaterloo.ca/~r3fraser/>

² Queen's University, Kingston, ON, Canada, K7L 3N6

janice@cs.queensu.ca

Abstract. The purpose of this work is to demonstrate that it is possible to cluster contact maps for pairs of alpha helices such that each of the clusters corresponds to a group of pairs of alpha helices with similar properties. The property of the configuration of helix pairs that was chosen for study is the packing attribute. The contact maps are compared to one another using a novel contact map comparison scheme based upon the locations of contacts in the contact maps. A k-nearest neighbours technique is used to perform the clustering, and the cosine between vectors corresponding to contact map regions was the distance metric. The clustering of contact maps to determine whether maps corresponding to similar packing values are placed into the same clusters yielded promising results.

1 Introduction

The aim of this research is to demonstrate that there are more tools available to augment those being used for protein structure prediction. The contact map is an abstract representation of the structure of a protein. People have attempted to recover the three dimensional structure of a protein from the contact map in the past [12], but the success has been limited. If specific properties of the three-dimensional structure of a protein may be predicted from the contact map, then we are given a useful tool. This prediction could be used in combination with other approaches to improve protein structure prediction techniques. Of course, the prediction of contact maps from amino acid sequences are in the early stages at present [5,10,11], but results are improving. For the purpose of this research, it is being assumed that the results of contact map prediction will be highly accurate at some later stage. A suitable challenge at present is to use an empirically determined contact map to determine the three-dimensional configuration of a pair of alpha helices.

^{*} This research has supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the PRECARN Institute for Robotics and Intelligent Systems (IRIS).

2 Contact Maps

This research pursues a slightly different tack from the conventional, by using contact maps as a fundamental tool. A contact map is an $N \times N$ matrix, where N is the number of amino acids in the given protein, and entry C_{ij} in the matrix is a boolean, indicating whether amino acid i is in contact with amino acid j . A threshold distance between atoms is the conventional definition of a contact; our group uses 10\AA between C_α atoms, as for our purposes it provides the most useful information. If the threshold is too high, more contacts will be introduced into the map, and many of the contacts will be meaningless for predictive purposes. If there are too few contacts, there is insufficient information for clustering.

Since we are working with pairs of alpha helices rather than entire proteins, it is necessary to isolate the contact maps associated with a particular pair of proteins. Further, we will be interested in isolating the interface region of the contact map for the pair as this is where potential interaction is occurring between the helices. This concept is shown in Figure 1.

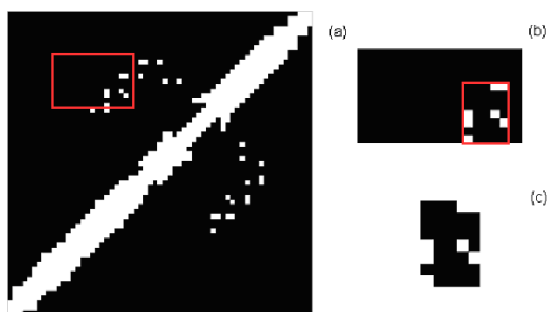


Fig. 1. (a) This is the contact map for protein 1a0a from the Protein Data Bank (PDB). The red rectangle indicates the area occupied by two helices, shown in (b). The contact map represents all of the amino acids for one alpha helix along the vertical axis and the other along the horizontal. This has been further refined to the interface area, shown in (c). The contact map interface is found by isolating the smallest rectangle containing all of the contact points from the contact map for the helix pair.

3 Methods

To test the hypothesis, we must first determine how we are to compare contact maps, as we need a distance metric for the comparisons. This problem is a field of study aside from the task at hand, so only a brief outline of the challenges will be given here. Once we have the comparison method, we can perform clustering and determine whether clusters of similar contact maps correspond to groups of helix pairs with similar packing attributes.

3.1 Distance Metrics

To begin the discussion, assume for the time being that we are comparing apples to apples: the contact maps that we are comparing are all of the same size and of similar orientations. If we have two contact maps C_1 and C_2 , then similar values at $C_1(i, j)$ and $C_2(i, j)$ indicate a similarity between the maps. If the maps we are comparing are of size k by l , the maps may also be represented as strings or vectors of length $n = k \times l$. We have the choice of many distance metrics for comparing these strings. The naïve choice would be to use Euclidean distance, which is the 2 norm distance between the strings. However, there are other metrics that are better suited to data of this type [3]. Euclidean distance has the property that true values and false values for attributes carry the same weight, and this does not work well for contact maps, which are generally sparse. Ertöz et al. [3] assert that when dealing with sparse data sets in high dimensions, it is often the case that the presence of an attribute is more significant than its absence. Therefore, the metrics chosen to measure distance should reflect this property. The Jaccard distance (or Jaccard coefficient or Jaccard index) is given by [7]:

$$d_J = \frac{C_{11}}{C_{11} + C_{10} + C_{01}},$$

where C_{11} is the number of contacts shared by both contact maps, C_{10} is the number of contacts in the first map not found in the second, and vice versa for C_{01} . Another metric with similar properties to the Jaccard distance is the cosine distance, which is given by the dot product between the normalized vectors corresponding to each contact map:

$$d_{cos} = \frac{C_1 \cdot C_2}{\|C_1\| \|C_2\|}.$$

Of course, contact maps from pairs of alpha helices are seldom the same size, and so we must manipulate the maps so that we can use these distance metrics. The comparison of contact maps involves first aligning the contact maps, so that a contact in one map has a similar meaning to a contact in another map. The proper alignment of contact maps is an open research problem, known as the Contact Map Overlap (CMO) problem [1]. Different visual techniques may be used to measure the similarity of the maps; these are discussed in detail in [9]. Another approach is to use graph theory and Lagrangian relaxation [1]. The nature of the data being used in the present application permits a simplified approach, since we are not dealing with contact maps for entire proteins. The challenge is to properly align the contact maps; this problem yielded a solution which involves dividing the contact maps into sub-classes.

3.2 Contact Map Classes

The easiest maps to align will be those maps with contacts in a corner of the map, as the corner provides a natural alignment point. Therefore, the first class

of contact map is any map with a contact in a corner of the map. For this application, the corners were treated as 2×2 areas in each corner. This makes sense, since this corresponds to the end of the helix. At the third residue, a turn around the axis is nearly complete, so it is not really at the end of the helix. However arbitrary, this threshold is effective, as will be demonstrated shortly.

The next class of contact maps is edge contacts; specifically any that have a contact within the outer two rows or columns around the perimeter of the contact map, but not in a corner. These maps can be aligned by the contacts present in the perimeter area, based upon the center of mass of the contacts in the perimeter.

Finally, the third class is the central contacts, which would be those maps that have no contacts within the outer perimeter. These maps are fairly difficult to align. The maps could be aligned again by their center of masses, but with two dimensions now factoring into the calculation, there is more room for error. Alternatively, one of the tools mentioned earlier for solving the contact map alignment problem could be used. Fortunately, the alignment of these maps is not an issue for the present problem, as will be shown. This classification process is performed greedily in the order presented here (corner \rightarrow edge \rightarrow central), so that any map belonging to a corner class is placed there first. Instances where maps could potentially belong to both the corner and edge classes are thus all considered instances of the corner class. Once all corner and edge maps have been removed from the source list, the central maps are what remain. Figure 2 shows the different classes of contact map schematically, and examples of each class are provided in Figure.

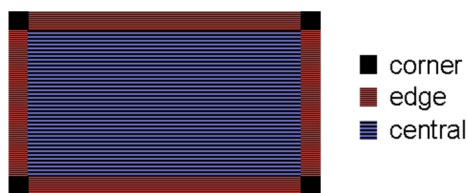


Fig. 2. The different classes of contact map for comparison. Corner contacts have contacts in a 2×2 corner of the map. Edge contacts have contacts along the perimeter of the map, but not in a corner. Central contacts have no contacts in the outer two rows or columns of the map.

3.3 Determination of Packing Values

The packing values for the pairs of alpha helices are found by first determining the axis for each helix using a rotational least squares method. Next, the packing attribute is found by locating the points of closest approach on each axis by applying the rotations calculated previously to both helices. If the line of closest approach between the axes is perpendicular to both axes, then there is packing. Otherwise, the packing attribute is considered false. This algorithm is presented in detail in [6].

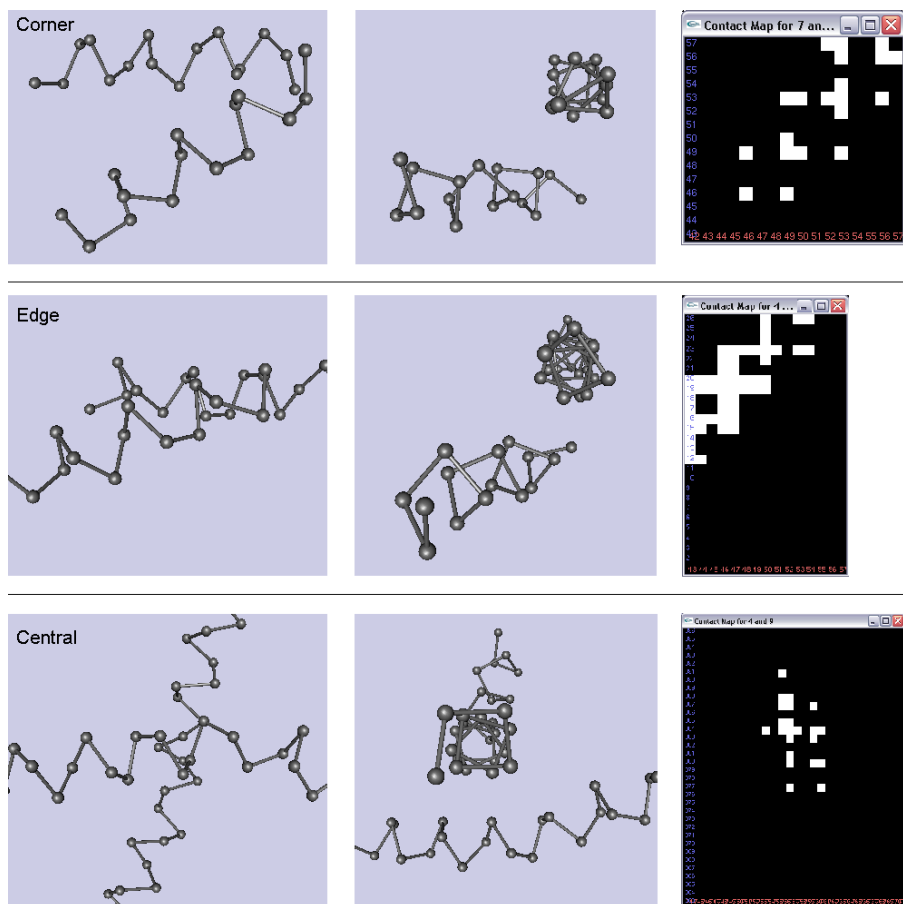


Fig. 3. Each row of the figure illustrates a pair of helices characteristic of each class. The left image is looking down the line of closest approach, the middle image shows the view down the axis of one of the helices, and the right image is the contact map for the pair at 10Å. The top row is an example of the corner class, notice that the closest points between the helices are at the ends of each. In this case, the helices are in an acute conformation; the opposite case would be contact at the end of each helix and they are angled away from each other. The middle row illustrates an example of the edge class, where the end of one helix is in contact with the middle of the other. This is a special case where both helices have such a contact. The bottom row shows a pair of helices in the central class. The helix pairs used from the top are helices 4 and 7 from PDB file 1A0A, 4 and 5 from 1A0A, and 4 and 9 from 1A28. All graphics were produced using the Hippy visualization software package [4].

4 Source Data

The data used for this study consists of 1078 pairs of alpha helices, collected from the Protein Data Bank (PDB) files of 171 proteins. The files were all alpha-helix only proteins with low homology. These helix pairs were selected such that all helices were of length at least 6, and that the contact interface for the pair was at least 2×2 . The central class contained 112 contact maps, and all maps corresponded to packing pairs of helices. Since all instances have a true value for packing, we can consider this to be one class. Since all instances have the same packing value, no further analysis is necessary. Both of the edge and corner classes contain instances of both values for the packing however, so clustering may be necessary.

The edge contact maps are challenging since there is no clear point of origin to use for the alignment step. However, as with the central maps, the task is not required for this application. We obtain 535 contact maps in this class. Of these maps, only 4 do not correspond to packing helix pairs, so it is fair to decide that these maps are outliers. Thus, contact maps with edge contacts generally correspond to packing pairs of alpha helices.

The corner maps, as mentioned earlier, are easy to align as they share a common origin. In addition, each contact map can be used as two data points since they can be reflected about a diagonal line through the origin. We can then transform each corner contact map such that all have a common origin through a series of flips. By these operations, we obtain a data set of 862 contact maps oriented in the corner. 794 of these maps correspond to packing pairs of helices, while 68 correspond to non-packing pairs. Now we have to modify the contact maps so they are all the same size. This has been accomplished in this study by simply taking the 15×15 square map rooted in the corner containing the contacts. This size has been chosen arbitrarily, but it is large enough to include most contacts for each map. If the map is smaller than this square in either dimension, the map is simply padded with zeroes. Now we have a collection of aligned maps of the same size that may be compared with a distance metric. The nature of the test data is summarized in Table 1.

Table 1. The characteristics of the contact maps used for clustering are summarized here. For each class of contact map, the number of instances in the source data set is given, as well as the number of corresponding pairs of alpha helices that exhibit packing or otherwise.

| Contact Map Class | Total Instances | Packing | Non-packing |
|-------------------|-----------------|--------------|-------------|
| Central | 112 | 112 (100%) | 0 (0%) |
| Edge | 535 | 531 (99.3%) | 4 (0.7%) |
| Corner | 431 | 397 (92.1%) | 34 (7.9%) |
| Doubled Corner | 862 | 794 (92.1%) | 68 (7.9%) |
| All Maps | 1078 | 1040 (96.5%) | 38 (3.5%) |

5 Clustering Analysis

Now we may proceed to cluster the corner contact maps, and determine whether clusters of maps share the same value for the packing attribute. Throughout this discussion, a contact map will often be referred to as a point, since that is what the clustering algorithm is working with. A 15×15 contact map which has been converted to a vector of length 225 is now being treated as a point in 225 dimensional space. The algorithm to be used to perform the clustering is a shared nearest neighbour approach, as outlined by Ertöz et al. [3]. The implementation of the algorithm is presented step by step.

Step 1 - Construction of the similarity matrix. The similarity matrix was constructed using both the Jaccard coefficient and the cosine distance in turn. Both yielded similar results in the final clustering, so the cosine distance was arbitrarily selected as the measure to be used for the remainder of the study. Given n contact maps to be clustered, the similarity matrix S is size $n \times n$, where entry $S(i, j)$ is the cosine distance from contact map i to j .

Step 2 - Sparsification of the similarity matrix. The similarity matrix is sparsified using k nearest neighbours (k-nn) sparsification. k nearest neighbours is a simple concept: it is simply that for some value k , find the k points closest to each point using the chosen distance metric. This is accomplished by first finding the k smallest distances in each row of the similarity matrix; this provides the k-nn for each point. To sparsify, we next check the members of the lists for each point to ensure mutuality. Suppose that point j is a member of the k-nn list for point i . Now we check the k-nn list for j to see if i is a member. If it is not, we remove j from the k-nn list for point i . The selection of an appropriate value for k is a trial and error process; the effectiveness depends on the size of the data set, the nature of the data being clustered, and the number of desired clusters in the outcome. For the corner contact maps, values between 4 and 15 were tested, and 12 produced the best results.

Step 3 - Construction of the shared nearest neighbour graph. We use a weighting scheme introduced by Jarvis and Patrick [8] to determine how well connected points are. This is done by finding the number of nearest neighbours two points share and how well connected they are. The strength of a connection between two points i and j is given by:

$$str(i, j) = \sum_{p_1}^{p_t} (k + 1 - m) \times (k + 1 - n)$$

where

- $i_m = j_n$, ie. some third point p in both lists;
- t is the number of shared nearest neighbours.

This sum is over every instance of a shared nearest neighbour in the lists for the points i and j . The shared nearest neighbour graph is an $n \times n$ matrix Snn , where entry $Snn(i, j)$ is defined by $str(i, j)$.

Steps 4 through 8 - Formation of the clusters. The next step is to find the connectivity of each point, $con(i)$, which is found by taking the sum of the strength of all of its connections:

$$con(i) = \sum_{j=1}^n Snn(i, j)$$

The connectivity of a point is used to determine how the clusters form. Points that have a connectivity higher than some threshold are chosen as representative points, and are used to nucleate clusters. Connectivity values were normalized to values between 0 and 1, and 0.6 was found to produce good clustering in this study. In the original Ertöz et al. [3] study, they removed all points with connectivity values below a given threshold and removed all links from the connectivity graph with weights below another threshold to eliminate noise. For the purposes of this study, the effects of these noise removal steps were minimized as it was desirable to consider all data as good data. As a result, 0.001 was used as the threshold for both steps. Finally, the clusters are formed by taking the representative points and all of the points that they are connected to as clusters.

Table 2. The clustering results are listed. There were 11 clusters of packing pairs, 2 of non-packing pairs, and 5 were mixed.

| Type | Cluster Number | Packing | Non-Packing |
|--------------------|----------------|---------|-------------|
| Packing | 1 | 21 | 0 |
| | 2 | 21 | 0 |
| | 3 | 18 | 0 |
| | 4 | 26 | 0 |
| | 5 | 16 | 0 |
| | 6 | 20 | 0 |
| | 7 | 37 | 0 |
| | 8 | 16 | 0 |
| | 9 | 29 | 0 |
| | 10 | 8 | 0 |
| | 11 | 18 | 0 |
| Mixed | 12 | 8 | 4 |
| | 13 | 30 | 2 |
| | 14 | 83 | 4 |
| | 15 | 13 | 4 |
| | 16 | 20 | 4 |
| Non-Packing | 17 | 0 | 14 |
| | 18 | 0 | 14 |

6 Clustering Results

The best results, as mentioned previously, were obtained by using a value of 12 for k when using the corner contact maps data set. A summary of the clustering results are presented in Table 2.

7 Conclusions and Future Work

Since the results of the clustering analysis showed promise, it could be concluded that the packing configuration of a pair of alpha helices could be predicted from the contact map. This is the first study to demonstrate that properties of the three-dimensional structure of a pair of alpha helices correspond to patterns in contact maps.

It is worth exploring other properties of alpha helices to determine how well clustering may predict these properties as well. These properties include the interhelical or interaxial angle, and the interhelical or interaxial distance for the pair. Finally, it will be interesting to see how these findings may be used to improve the accuracy of protein structure prediction techniques.

References

1. A. Caprara and G. Lancia. Structural alignment of large-size proteins via Lagrangian relaxation. In *RECOMB '02: Proceedings of the sixth annual international conference on Computational biology*, pages 100–108, New York, NY, USA, 2002. ACM Press.
2. C. Chothia, M. Levitt, and D. Richardson. Helix to helix packing in proteins. *Journal of Molecular Biology*, 145:215–250, 1981.
3. L. Ertöz, M. Steinbach, and V. Kumar. A new shared nearest neighbor clustering algorithm and its applications. In *Proceedings of the Workshop on Clustering High Dimensional Data and its Applications, Second SIAM International Conference on Data Mining*, 2002.
4. R. Fraser and J. Glasgow. Introducing Hippy: A visualization tool for understanding the α -helix pair interface. In *Proceedings of the 2006 International Conference on Bioinformatics and Computational Biology (BIOCOMP)*, 2006.
5. P. Fariselli, O. Olmea, A. Valencia, and R. Casadio. Prediction of contact maps with neural networks and correlated mutations. *Protein Engineering*, 14(11):835–843, 2001.
6. R. Fraser. A Tale of Two Helices: A study of alpha helix pair conformations in three-dimensional space. Master's thesis, Queen's University, 2006.
7. P. Jaccard. Nouvelles recherches sur la distribution florale. *Bulletin de la Société Vaudoise des Sciences Naturelles*, 44:223–270, 1908.
8. R.A. Jarvis and E.A. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computers*, C22:1025–1034, 1973.
9. T. Kuo. A computational approach to contact map similarity. Master's thesis, Queen's University, 2005.
10. G. Pollastri and P. Baldi. Prediction of contact maps by GIOHMMs and recurrent neural networks using lateral propagation from all four cardinal corners. *Bioinformatics*, 18(Supplement 1):S62–S70, 2002.
11. M. Punta and B. Rost. PROFcon: novel prediction of long-range contacts. *Bioinformatics*, 21(13):2960–2968, 2005.
12. M. Vendruscolo, E. Kussell, and E. Domany. Recovery of protein structure from contact maps. *Folding and Design*, 2(5):295–306, 1997.

Dynamic Data Probes

David W. Pearson

Hubert Curien Laboratory - UMR CNRS 5516,
Jean Monnet University of Saint-Etienne,
18 rue Benoit Lauras, 42023 Saint-Etienne, France
`david.pearson@univ-st-etienne.fr`

Abstract. In this paper we look at a dynamic method for analysing data, called a data probe. The probe flies through the data space and is affected by the proximity and number of data points. The trajectory followed by the probe provides information about how the data are organised geometrically. We apply a state feedback method to the probe equations to make the probe search out certain data values.

1 Introduction

The inspiration for this work comes from an exiting new area of research called data sonification [2]. In data sonification sound is generated from data to add a new dimension to it and improve the data analysis task. We are not directly interested in generating sound from our data, but the model based method of data sonification presented in [2] provides the basis for our data probe.

We are particularly interested in data clusters [1], where data points are locally grouped together into clusters. Within a cluster, the data have some sort of homogeneous property, or, because we are dealing with real data, nearly homogeneous. We are working with data belonging to a Euclidian space, we do not consider the case of discrete data.

2 Data Probe Dynamics

The basic analogy is a comet flying through a solar system. If the comet doesn't fly near any planets then it will just fly on without changing its trajectory. However, if it flies near to a massive object like a planet then it will be deviated from its trajectory. If you are able to observe the trajectory, but not the planets, then you will be able to deduce the presence and size of the massive object by the comet's new trajectory. We consider data points to be massive objects and use Newton's equations of motion to describe the dynamics of the probe (comet). The full details of these equations are presented in [2].

We assume that the data are supplied in pairs (x_k, y_k) for $k = 1, \dots, m$, where the $x_k \in \mathbb{R}^n$ are taken to be inputs and the $y_k \in \mathbb{R}$ outputs. It is also assumed that there exists a mapping from inputs to outputs $\pi : \mathbb{R}^n \rightarrow \mathbb{R}$ that satisfies $\pi(x_k) = y_k$ for all the supplied data. The fundamental hypothesis is that the

data are organised into clusters and that for any x_i, x_j belonging to the same cluster then $\pi(x_i) = \pi(x_j) = y_0$, or what is more reasonable $\pi(x_i) \approx \pi(x_j) \approx y_0$.

We denote the data points as

$$d_k = \begin{bmatrix} y_k \\ x_k \end{bmatrix}$$

then we define the following attraction function

$$\phi(y, x) = \sum_{k=1}^m \exp(-\alpha \| \begin{bmatrix} y \\ x \end{bmatrix} - d_k \|^2) \tag{1}$$

where $\alpha > 0$ is a parameter related to the radius of influence of each data point.

The Newtonian equations of motion include acceleration, velocity and resistance terms and so we need to augment the dimension of the system from \mathbb{R}^{n+1} to $\mathbb{R}^{2(n+1)}$ in order to recover an autonomous set of differential equations. So, we define the following vector

$$z = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} y \\ x \\ \dot{y} \\ \dot{x} \end{bmatrix} \tag{2}$$

where $\dot{y} = \frac{dy}{dt}$ etc. Using (1) and (2) the equation of motion of the probe is then given by the following

$$\dot{z} = \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \\ \dot{z}_4 \end{bmatrix} = \begin{bmatrix} z_3 \\ z_4 \\ \nabla\phi(z_1, z_2) - \gamma \begin{bmatrix} z_3 \\ z_4 \end{bmatrix} \end{bmatrix} \tag{3}$$

where γ is a positive scalar parameter and

$$\nabla\phi(z_1, z_2) = \begin{bmatrix} \frac{\partial\phi}{\partial z^1} \\ \vdots \\ \frac{\partial\phi}{\partial z^{n+1}} \end{bmatrix}$$

note that the partial derivatives are up to $\frac{\partial}{\partial z^{n+1}}$ because, by definition, $x^n = z^{n+1}$. We note also that this equation is slightly different to the model presented in [2] because we have not included any masses, we assume that all the data points have unit mass.

3 Guiding the Probe

The system (3) is autonomous and the only parameters that can be changed are α, γ and the initial point $z(0)$. The objective that we seek to achieve in this paper is to get the probe to search out a particular value of the output variable

y . In this way, we want the probe to end up in a part of the data space where this particular value of the output variable is predominant, i.e. a cluster. To do this we introduce a feedback term in the system (3). First of all we define an output function for (3)

$$h(z) = z_1 - y_0 \tag{4}$$

y_0 is the target value for $z_1 = y$.

Clearly, to achieve our aim, from (4) we need to calculate a trajectory $z(t)$, $t > 0$ that will satisfy $h(z(t)) = 0$. In control theory parlance this is referred to as output zeroing [3]. In the simple output zeroing algorithm $z(0)$ has to be chosen to satisfy $h(z(0)) = 0$. However, we would like to have free choice of $z(0)$ to let the probe explore the space. For this reason, we apply the more general and powerful method of asymptotic output zeroing. This is where $z(t)$ converges to a trajectory that satisfies $h(z(t)) = 0$ even if the initial part of the trajectory doesn't.

We do not present the full details of the method in this paper, the interested reader is advised to consult [3]. Basically, we modify the system (3) by introducing a feedback term into the line corresponding to \dot{z}_3 by setting

$$\dot{z}_3 = \frac{\partial\phi(z_1, z_2)}{\partial z_1} - \gamma z_3 + u(z) \tag{5}$$

where $u(z)$ is the feedback term.

Following [3], it can be shown that setting the feedback term to the following value

$$u(z) = -\frac{\partial\phi(z_1, z_2)}{\partial z_1} + \gamma z_3 - c_0(z_1 - y_0) - c_1 z_3 \tag{6}$$

leads to the desired result. By this we mean that if we define an error function as follows

$$e(t) = h(z(t))$$

then $e(t)$ satisfies the following linear differential equation

$$c_1 \dot{e}(t) + c_0 e(t) = 0$$

and so it suffices to choose the values of the two parameters c_1 and c_0 so that $e(t) \rightarrow 0$. This will be the case, for example, if we set $c_1 = 1$ and $c_0 > 0$.

4 Simulation

We produced some simulated data by calculating 20 examples of each of three clusters based on the points $[1 \ 1 \ 1]^T$, $[-1 \ 1 \ -1]^T$ and $[0 \ -1 \ -1]^T$ plus random noise. For our preliminary investigations we set $\alpha = \gamma = 0.5$ in (3) and chose an initial point $z(0) = [1.5 \ 1.5 \ 1.5 \ 0 \ 0]^T$. First of all we integrated the system (3)

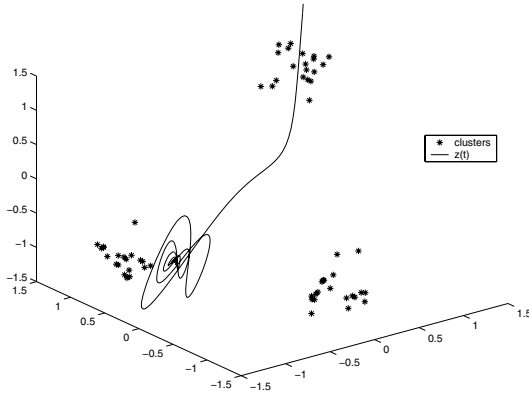


Fig. 1. Trajectory without feedback

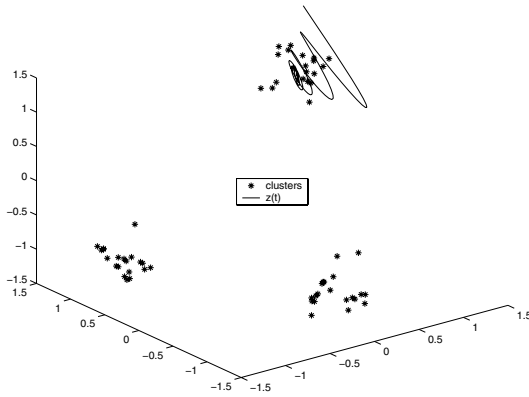


Fig. 2. Trajectory with $y_0 = 1$

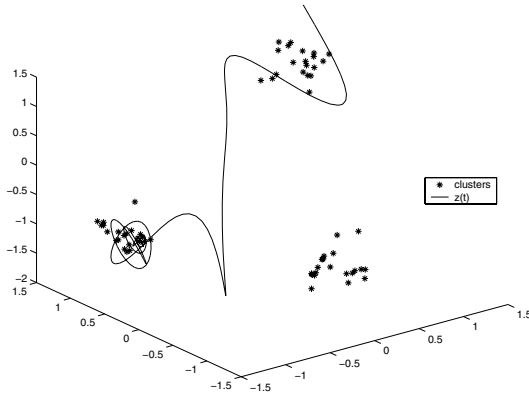


Fig. 3. Trajectory with $y_0 = -1$

using a standard Runge-Kutta routine with no feedback term, i.e. with $u = 0$ in (6). The result can be seen in figure 1, note that we only plot the first three coordinates of $z(t)$, i.e. $(y(t), x(t))$ but not the derivatives. The probe circles around and converges to a point near to a cluster, but not exactly a cluster centre.

Then, we brought in the feedback term by setting $c_1 = 1$, $c_0 = 0.5$ in (6) and integrating (3) this time with feedback. We carried out three trial runs by setting successively $y_0 = 1$, $y_0 = -1$ and $y_0 = 0$ (the three theoretical output values of the clusters). The results can be seen in figures 2, 3 and 4. We can see that the trajectories encircle the clusters before converging to the centre of the cluster in each case.

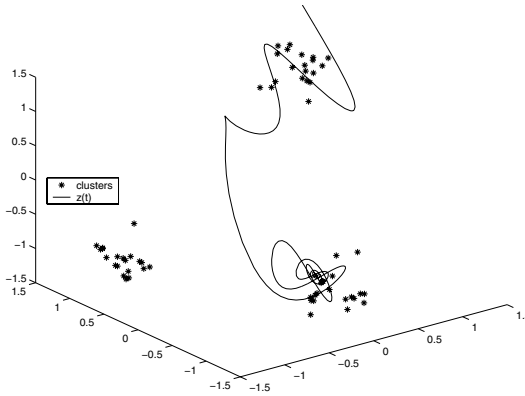


Fig. 4. Trajectory with $y_0 = 0$

5 Conclusion

In this paper, we have presented a method of data analysis based on a dynamic data probe. By combining this method with another coming from control theory, we have shown that it is feasible to guide the probe to seek out particular output values and thus to seek out particular data clusters. We are currently investigating the influence of the various parameters in our model.

References

1. Chiu, S.L.: Fuzzy model identification based on cluster estimation. *Journal of Intelligent and Fuzzy Systems* **2** (1995) 267–278
2. T. Hermann, T. and Ritter, H.: Listen to your Data: Model-Based Sonification for Data Analysis. *Advances in intelligent computing and multimedia systems*, M. R. Syed (editor), International Institute for Advanced Studies in System Research and Cybernetics (1999) 189–194
3. Isidori, A.: *Nonlinear Control Systems*. third edition, Springer-Verlag (1995)

Classifying Chemical Compounds Using Contrast and Common Patterns

Andrzej Dominik, Zbigniew Walczak, and Jacek Wojciechowski

Warsaw University of Technology, Institute of Radioelectronics
Nowowiejska 15/19, 00-665 Warsaw, Poland
A.Dominik@elka.pw.edu.pl, Z.Walczak@elka.pw.edu.pl,
J.Wojciechowski@ire.pw.edu.pl

Abstract. The problem of classifying chemical compounds is studied in this paper. An approach based on minimal contrast and common topological patterns discovered from compounds dataset is presented. The algorithm is strongly associated with the classical emerging patterns techniques known from decision tables. We tested the proposed algorithm on real classification problems. Results show that it provides better classification accuracy than other existing algorithms. Another advantage of the introduced classifier is that it has a simple, understandable structure and can be easily extended by the expert knowledge.

1 Introduction

Classification problems have been deeply researched due to variety of applications. They appear in different fields of science and industry and may be solved using different techniques: e.g. neural networks, rough sets, etc. Sample applications in medicine/chemistry may include detecting mutagenicity or carcinogenicity of chemical compounds for a given organism.

Chemical molecules have various representations depending on their dimensions and features. Basic representations are: (1D) strings expressed in SMILES language, (2D) topological graphs of atoms and bonds, (3D) geometrical structures containing coordinates of atoms. This paper concentrates on 2D topological graphs (atoms correspond to labelled vertices and bonds to labelled edges). These graphs are typically quite small (in terms of number of vertices and edges) and the average number of edges per vertex is usually slightly above 2.

The problem of classifying chemical compounds has been recently deeply studied. Two major approaches have been developed. The first one, quantitative structure-activity relationships (QSAR) concentrates on physico-chemical properties derived from compounds while the other one searches directly structure of the compound. The former requires genuine chemical knowledge while the latter does not. Moreover the second approach is not limited to chemical applications only: it was successfully applied in other areas (eg. cellphone reviews classification). Its main idea is to discover small, significant substructures (patterns, fragments) from the original structures which discriminate between different classes.

Many techniques were used to achieve this goal, like Support Vector Machines or Frequent Patterns.

This paper makes a few important contributions. The concepts of contrast and common subgraphs are extended and used for building a Contrast Common Patterns Classifier (CCPC). Some typical emerging patterns ideas are adapted to improve classification results. Results for real chemical compounds classification problems obtained by using the considered classifier are provided.

2 Preliminary Terminology

In this section we introduce some basic concepts and definitions [18], [3] that will be used in the subsequent sections.

All discussed graphs are assumed to be undirected, connected (any two vertices are linked by a path), labelled (both vertices and edges possess labels) and simple (without loops and parallel edges). By the size of a graph we mean the number of its edges. Capital letters (G, S, \dots) denote single graphs while calligraphic letters ($\mathcal{G}, \mathcal{N}, \mathcal{P}, \dots$) denote sets of graphs.

Definition 1. *Labelled graph G is a quadruple (V, E, α, β) , where V is a non-empty finite set of vertices, $E \subseteq V \times V$ is a non-empty finite set of edges and α, β are functions assigning labels to vertices and edges, respectively.*

Definition 2. *A graph $S = (W, F, \alpha, \beta)$ is a subgraph of $G = (V, E, \alpha, \beta)$ (written as $S \subseteq G$) if: (1) $W \subseteq V$ and (2) $F \subseteq E \cap (W \times W)$.*

Definition 3. *Let \mathcal{G} be a set of graphs, $G' = (V', E', \alpha', \beta')$ and $G = (V, E, \alpha, \beta)$. We say that G' is isomorphic to G (written as $G' \simeq G$) if there is an injective function $f: V' \rightarrow V$ such that: (1) $\forall e' = (u', v') \in E' \exists e = (f(u'), f(v')) \in E$, (2) $\forall u' \in V', \alpha'(u') = \alpha(f(u'))$ and (3) $\forall e' \in E', \beta'(e') = \beta(f(e'))$. If $f: V' \rightarrow V$ is a bijective function then G' is automorphic to G (written as $G' = G$). If G' is not isomorphic to G then we write $G' \not\simeq G$.*

A graph G' is \mathcal{G} -isomorphic (written as $G' \simeq \mathcal{G}$) if: (1) $\exists G \in \mathcal{G} : G' \simeq G$. A graph G' is not \mathcal{G} -isomorphic (written as $G' \not\simeq \mathcal{G}$) if: (1) $\forall G \in \mathcal{G} : G' \not\simeq G$.

Definition 4. *Given the sets of graphs \mathcal{P}, \mathcal{N} and a graph $M_{\mathcal{P}\mathcal{N}}$. $M_{\mathcal{P}\mathcal{N}}$ is a common subgraph for \mathcal{P} and \mathcal{N} if: (1) $M_{\mathcal{P}\mathcal{N}} \simeq \mathcal{P}$ and (2) $M_{\mathcal{P}\mathcal{N}} \simeq \mathcal{N}$. Set of all common subgraphs for \mathcal{P} and \mathcal{N} will be denoted by $\mathcal{M}_{\mathcal{P}\mathcal{N}}$. Set of all minimal (with respect to size i.e. containing only one edge and two vertices) common subgraphs for \mathcal{P} and \mathcal{N} will be denoted as $\mathcal{M}_{\mathcal{P}\mathcal{N}}^{\text{Min}}$.*

Definition 5. *Given the sets of graphs \mathcal{N} and a graph P . A graph $C_{P \rightarrow \mathcal{N}}$ is a contrast subgraph of P with respect to \mathcal{N} if: (1) $C_{P \rightarrow \mathcal{N}} \simeq P$ and (2) $C_{P \rightarrow \mathcal{N}} \not\simeq \mathcal{N}$. It is minimal (with respect to isomorphism) if all of $C_{P \rightarrow \mathcal{N}}$'s strict subgraphs are not contrast subgraphs. Set of all minimal contrast subgraphs of P with respect to \mathcal{N} will be denoted as $\mathcal{C}_{P \rightarrow \mathcal{N}}^{\text{Min}}$.*

Definition 6. Given the sets of graphs $\mathcal{P} = \{P_1, \dots, P_n\}$ and \mathcal{N} . Let $\mathcal{C}_{\mathcal{P}_i \rightarrow \mathcal{N}}^{\text{Min}}$ be the set of all minimal contrast subgraphs of P_i with respect to \mathcal{N} , where $i \in \langle 1, n \rangle$. $\mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}}$ is a set of all minimal contrast subgraphs of \mathcal{P} with respect to \mathcal{N} if: (1) $\forall C \in \mathcal{C}_{\mathcal{P}_i \rightarrow \mathcal{N}}^{\text{Min}} \exists J \in \mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}} : J \simeq C$, for $i \in \langle 1, n \rangle$, (2) $\forall J_1 \in \mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}} \neg \exists J_2 \in \mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}} \setminus J_1 : J_2 \simeq J_1$.

$\mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}}$ contains all minimal subgraphs (patterns) which are present in \mathcal{P} (i.e. each subgraph in $\mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}}$ is isomorphic to at least one graph from \mathcal{P}) and are not present in \mathcal{N} (i.e. each subgraph in $\mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}}$ is not isomorphic to any graph from \mathcal{N}). What is more $\mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}}$ contains only minimal (with respect to size and isomorphism) subgraphs.

Definition 7. Given the sets of graphs \mathcal{G} , \mathcal{N} , \mathcal{P} and a graph G . Let $\mathcal{S} = \{G' \in \mathcal{G} : G \simeq G'\}$. Support of graph G in \mathcal{G} is defined as follows: $\text{supp}_{\mathcal{G}}(G) = \frac{\text{card}(\mathcal{S})}{\text{card}(\mathcal{G})}$, where $\text{card}(\mathcal{G})$ denotes the cardinal number of set \mathcal{G} . Growth rate of graph G in favour of \mathcal{P} against \mathcal{N} is expressed as follows: $\rho_{\mathcal{P} \rightarrow \mathcal{N}}(G) = \frac{\text{supp}_{\mathcal{P}}(G)}{\text{supp}_{\mathcal{N}}(G)}$.

3 Related Work

In this section we review the state of the art in the areas associated with mining contrast and common patterns.

Contrast patterns are substructures that appear (appear frequently) in one class of objects and don't appear (appear infrequently) in other classes. In data mining patterns which uniquely identify certain class of objects are called jumping emerging patterns (JEP). Patterns common for different classes are called emerging patterns (EP). Concepts of jumping emerging patterns and emerging patterns [11], [4], [5] have been deeply researched as a tool for classification purposes.

The concept of contrast subgraphs was studied in [18], [1]. Ting and Bailey [18] proposed an algorithm (containing backtracking tree and hypergraph traversal algorithm) for mining all disconnected contrast subgraphs from dataset.

Another relevant area to review is mining frequent structures. Frequent structure is a structure which appears in samples of a given dataset more frequently than the specified threshold. Agarwal and Srikant proposed an efficient algorithm for mining frequent itemsets in the transaction database called Apriori. Similar algorithms were later proposed for mining frequent subgraphs from graphs dataset: [9], [12]. They were also used for the classification purposes [2].

Mining patterns in graphs dataset which fulfil given conditions is a much more challenging task than mining patterns in decision tables (relational databases). The most computationally complex tasks are isomorphism and automorphism. The first problem is proved to be *NP*-complete while the complexity of the other one is still not known. All the algorithms for solving the isomorphism problem present in the literature, have an exponential time complexity in the worst case but polynomial solution has not been yet disproved. A universal exhaustive algorithm for both of these problems was proposed in [19]. It operates

on the matrix representation of graphs and tries to find a proper permutation of nodes. Search space can be greatly reduced by using nodes invariants and iterative partitioning [7]. Moreover multiple graph isomorphism problems can be efficiently performed with canonical labelling [14], [7]. Canonical label is a unique representation (code) of a graph such that two isomorphic graphs have the same canonical label.

Another important issue is generating all non-isomorphic subgraphs of a given graph. The algorithm for generating DFS (Depth First Search) code [20] can be used to enumerate all subgraphs and reduce the number of required isomorphism checking.

One of the most popular approaches for graph classification is based on SVM (Support Vector Machines). SVMs have good generalization properties (both theoretically and experimentally) and they operate well in high-dimensional datasets. Numerous different kernels using all three compound representations were designed for this method [17], [10].

4 Classifier

In this section we propose a classification algorithm called: CCPC (Contrast Common Patterns Classifier). We present only the general concept without implementation details.

The concept of contrast subgraph is directly associated with the concept of jumping emerging pattern (JEP). They both define a pattern (either subgraph or set of items) exclusive for one class of objects. Similarly, the common subgraphs are associated with emerging patterns (EP) i.e. patterns that are present in both classes of objects. Measures for classical emerging patterns designed for classification purposes are mainly based on the support of a pattern in different classes of objects. This section adapts some classical scoring schemes to be used with contrast and common subgraphs.

Let \mathcal{G} be a set of training graphs (graphs used for the learning of a classifier) and G be a test graph (graph to classify). Let \mathcal{G} be divided into two decision disjoint classes: positive \mathcal{P} and negative \mathcal{N} , $\mathcal{G} = \mathcal{N} \cup \mathcal{P}$. Let $\mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}}$ be the set of all minimal contrast subgraphs of \mathcal{P} with respect to graph set \mathcal{N} and $\mathcal{C}_{\mathcal{N} \rightarrow \mathcal{P}}^{\text{Min}}$ be the set of all minimal contrast subgraphs of \mathcal{N} with respect to \mathcal{P} . Let $\mathcal{M}_{\mathcal{P}\mathcal{N}}^{\text{Min}}$ be the set of all minimal common subgraphs for \mathcal{P} and \mathcal{N} .

Let us now define a few score routines used for classification. Score is obtained using contrast subgraphs according to the following equations:

$$\text{scConA}_{\mathcal{P}}(G) = \sum_{K \in \mathcal{K}} \text{supp}_{\mathcal{P}}(K), \quad \mathcal{K} = \{K : K \in \mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}} \wedge K \simeq G\} \quad (1)$$

$$\text{scConA}_{\mathcal{N}}(G) = \sum_{K \in \mathcal{K}} \text{supp}_{\mathcal{N}}(K), \quad \mathcal{K} = \{K : K \in \mathcal{C}_{\mathcal{N} \rightarrow \mathcal{P}}^{\text{Min}} \wedge K \simeq G\} \quad (2)$$

$$\text{scConB}_{\mathcal{P}}(G) = \frac{1}{\lambda_{\mathcal{P}}} * \sum_{K \in \mathcal{K}} \text{supp}_{\mathcal{P}}(K), \quad \mathcal{K} = \{K : K \in \mathcal{C}_{\mathcal{P} \rightarrow \mathcal{N}}^{\text{Min}} \wedge K \simeq G\} \quad (3)$$

$$\text{scConB}_{\mathcal{N}}(G) = \frac{1}{\lambda_{\mathcal{N}}} * \sum_{K \in \mathcal{K}} \text{supp}_{\mathcal{N}}(K), \quad \mathcal{K} = \{K : K \in \mathcal{C}_{\mathcal{N} \rightarrow \mathcal{P}}^{\text{Min}} \wedge K \simeq G\} \quad (4)$$

where $\lambda_{\mathcal{P}}, \lambda_{\mathcal{N}}$ are scaling factors. They are median values from statistics of the contrast scores (1) (2) determined for each graph from both classes: \mathcal{P}, \mathcal{N} .

Score is also calculated using common subgraphs according to the following equations:

$$\text{scComA}_{\mathcal{P}}(G) = \sum_{K \in \mathcal{K}} \text{supp}_{\mathcal{P}}(K), \quad \mathcal{K} = \{K : K \in \mathcal{M}_{\mathcal{P}\mathcal{N}}^{\text{Min}} \wedge K \simeq G\} \quad (5)$$

$$\text{scComA}_{\mathcal{N}}(G) = \sum_{K \in \mathcal{K}} \text{supp}_{\mathcal{N}}(K), \quad \mathcal{K} = \{K : K \in \mathcal{M}_{\mathcal{P}\mathcal{N}}^{\text{Min}} \wedge K \simeq G\} \quad (6)$$

$$\text{scComB}_{\mathcal{P}}(G) = \sum_{K \in \mathcal{K}} \rho_{\mathcal{P} \rightarrow \mathcal{N}}(G), \quad \mathcal{K} = \{K : K \in \mathcal{M}_{\mathcal{P}\mathcal{N}}^{\text{Min}} \wedge K \simeq G\} \quad (7)$$

$$\text{scComB}_{\mathcal{N}}(G) = \sum_{K \in \mathcal{K}} \rho_{\mathcal{N} \rightarrow \mathcal{P}}(G), \quad \mathcal{K} = \{K : K \in \mathcal{M}_{\mathcal{P}\mathcal{N}}^{\text{Min}} \wedge K \simeq G\} \quad (8)$$

In (5) and (6) score depends directly on the support of the subgraphs, whereas in (7) and (8) score depends on the growth rate of certain patterns.

Classification process looks as follows. First scores based on contrast subgraphs are calculated for each class. We can choose between presented scoring schemes:

- scConA - we calculate $\text{scConA}_{\mathcal{P}}(G)$ and $\text{scConA}_{\mathcal{N}}(G)$ from eqs. (1) and (2),
- scConB - we calculate $\text{scConB}_{\mathcal{P}}(G)$ and $\text{scConB}_{\mathcal{N}}(G)$ from eqs. (3) and (4).

Test example G is assigned to a class with a higher score. If both scores are equal then G remains unclassified and scores based on common subgraphs are calculated. Again we can choose one of the two approaches:

- scComA - we calculate $\text{scComA}_{\mathcal{P}}(G)$ and $\text{scComA}_{\mathcal{N}}(G)$ eqs (5) and (6),
- scComB - we calculate $\text{scComB}_{\mathcal{P}}(G)$ and $\text{scComB}_{\mathcal{N}}(G)$ eqs (7) and (8).

Test sample is assigned to a class with a higher score. If both scores are equal then G remains unclassified.

5 Experiments

We performed a set of experiments on two popular chemical compound datasets (Table 1) to measure performance and properties of our classification algorithms. Later we compared the results with those achieved by other approaches.

Dataset MUTAG [16] reports mutagenicity of 188 chemical compounds in *Salmonella typhimurium*. PTC (Predictive Toxicology Challenge) [15], [8] reports carcinogenicity of several hundred compounds for female mice (FM), female rats (FR), male mice (MM) and male rats (MR). In the case of mutagenicity one

classification problem is considered while in the PTC four ones, i.e. for FM, FR, MM, MR. In each problem there are two decision classes: negative and positive.

The performance of classifiers (ability to assign the correct class to a compound) was evaluated using leave-one-out cross-validation procedure. Accuracy of a classifier is expressed as the percentage of correctly classified molecules. Additionally we plotted ROC curves [13] which display performance of classifiers

Table 1. Detailed information on MUTAG and PTC datasets

| Dataset | Number of compounds (%) | | | Average number of | | Max. number of | |
|---------|-------------------------|------------|------------|----------------------|-------|----------------|-------|
| | Total | Negative | Positive | Atoms | Bonds | Atoms | Bonds |
| | | | | in a single compound | | | |
| MUTAG | 188 | 63 (33.5) | 125 (66.5) | 17.9 | 19.8 | 28 | 33 |
| PTC-FM | 349 | 206 (59.0) | 143 (41.0) | 14.1 | 14.5 | 64 | 71 |
| PTC-FR | 351 | 230 (65.5) | 121 (34.5) | 14.6 | 15.0 | 64 | 71 |
| PTC-MM | 336 | 207 (61.6) | 129 (38.4) | 14.0 | 14.4 | 64 | 71 |
| PTC-MR | 344 | 192 (55.8) | 152 (44.2) | 14.3 | 14.7 | 64 | 71 |

Table 2. Average cardinality of contrast and common subgraphs sets for different datasets (MUTAG, PTC) and maximal contrast subgraph sizes (1, 4, 7(8)). Average calculated over all iterations performed during leave-one-out cross-validation procedure.

| Average card. of set | MUTAG | | | PTC-FM | | | PTC-FR | | | PTC-MM | | | PTC-MR | | |
|------------------------------------|-------|----|-----|--------|-----|-----|--------|-----|-----|--------|-----|-----|--------|-----|-----|
| | 1 | 4 | 8 | 1 | 4 | 7 | 1 | 4 | 7 | 1 | 4 | 7 | 1 | 4 | 7 |
| $C_{N \rightarrow P}^{\text{Min}}$ | 2 | 38 | 272 | 19 | 182 | 946 | 29 | 187 | 944 | 23 | 184 | 910 | 13 | 156 | 849 |
| $C_{P \rightarrow N}^{\text{Min}}$ | 2 | 39 | 550 | 11 | 97 | 589 | 3 | 88 | 572 | 11 | 86 | 585 | 14 | 106 | 659 |
| $\mathcal{M}_{PN}^{\text{Min}}$ | 14 | 14 | 14 | 32 | 32 | 32 | 32 | 32 | 32 | 29 | 29 | 29 | 31 | 31 | 31 |

Table 3. Leave-one-out cross-validation results (accuracy and ROC score in %) for the MUTAG dataset using different contrast and common subgraph scorings (scConA, scConB, scComA, scComB) with maximal contrast subgraph sizes (1 - 8). Bold face indicates highest values of accuracy and ROC score for each dataset

| Dataset (Maximal contrast subgraph size) | scConA scComA | | scConA scComB | | scConB scComA | | scConB scComB | |
|--|------------------|------|------------------|------|------------------|------|------------------|-------------|
| | Acc. | ROC | Acc. | ROC | Acc. | ROC | Acc. | ROC |
| MUTAG | | | | | | | | |
| 1 | 17.0 | 0.0 | 74.5 | 62.7 | 17.0 | 0.0 | 74.5 | 62.7 |
| 2 | 19.1 | 26.6 | 76.1 | 65.1 | 19.1 | 26.6 | 76.1 | 65.1 |
| 3 | 27.7 | 34.2 | 78.2 | 70.6 | 28.2 | 34.6 | 78.7 | 71.0 |
| 4 | 33.0 | 40.2 | 83.0 | 79.3 | 36.7 | 43.7 | 86.7 | 82.9 |
| 5 | 47.3 | 52.5 | 84.0 | 81.7 | 51.6 | 56.1 | 88.3 | 85.3 |
| 6 | 72.3 | 72.9 | 87.2 | 84.5 | 76.6 | 77.3 | 91.5 | 88.9 |
| 7 | 80.9 | 76.5 | 84.6 | 80.5 | 87.8 | 84.9 | 91.5 | 88.9 |
| 8 | 81.9 | 76.6 | 84.0 | 79.3 | 90.4 | 87.3 | 92.6 | 90.1 |

without regard to class distribution. The ROC curve plots the number of true positive predictions as a function of the number of false positive predictions, changing together with the change of a chosen parameter of the method. ROC score is expressed as a normalized area under the ROC curve.

CCPC classifier is a discrete classifier (it generates only class label for each test object). Such a classifier is characterized by a pair (false positive rate and

Table 4. Leave-one-out cross-validation results (accuracy and ROC score in %) for the PTC dataset using different contrast and common subgraph scorings (scConA, scConB, scComA, scComB) with maximal contrast subgraph sizes (1 - 7). Bold face indicates highest values of accuracy and ROC score for each dataset.

| Dataset (Maximal contrast subgraph size) | scConA scComA | | scConA scComB | | scConB scComA | | scConB scComB | |
|--|------------------|------|------------------|------|------------------|-------------|------------------|-------------|
| | Acc. | ROC | Acc. | ROC | Acc. | ROC | Acc. | ROC |
| PTC-FM | | | | | | | | |
| 1 | 61.3 | 54.5 | 41.3 | 45.2 | 61.6 | 54.9 | 41.5 | 45.6 |
| 2 | 60.7 | 54.2 | 43.3 | 46.5 | 62.2 | 55.8 | 44.7 | 48.0 |
| 3 | 59.9 | 53.6 | 48.4 | 50.0 | 62.8 | 56.7 | 51.3 | 53.1 |
| 4 | 61.9 | 56.4 | 59.0 | 58.7 | 66.8 | 61.6 | 63.9 | 63.9 |
| 5 | 61.3 | 56.3 | 61.0 | 59.3 | 69.6 | 65.1 | 69.3 | 68.0 |
| 6 | 59.6 | 54.8 | 58.7 | 56.0 | 73.1 | 69.2 | 72.2 | 70.4 |
| 7 | 59.9 | 55.8 | 58.7 | 56.0 | 79.1 | 76.6 | 77.9 | 76.8 |
| PTC-FR | | | | | | | | |
| 1 | 62.4 | 49.8 | 36.2 | 42.9 | 63.2 | 50.6 | 37.0 | 43.7 |
| 2 | 63.0 | 51.0 | 36.8 | 43.9 | 64.1 | 52.2 | 37.9 | 45.2 |
| 3 | 64.7 | 53.3 | 46.7 | 51.9 | 65.8 | 54.5 | 47.9 | 53.2 |
| 4 | 66.4 | 55.7 | 56.4 | 57.3 | 71.8 | 62.0 | 61.8 | 63.6 |
| 5 | 65.0 | 54.9 | 57.5 | 56.1 | 73.8 | 64.9 | 66.4 | 66.1 |
| 6 | 62.7 | 53.9 | 58.4 | 55.9 | 74.4 | 66.7 | 70.1 | 68.8 |
| 7 | 63.2 | 55.7 | 59.5 | 56.6 | 78.6 | 71.9 | 74.9 | 72.8 |
| PTC-MM | | | | | | | | |
| 1 | 62.5 | 57.2 | 36.6 | 40.8 | 62.8 | 57.4 | 36.9 | 41.1 |
| 2 | 61.3 | 56.2 | 39.0 | 41.7 | 62.2 | 56.9 | 39.9 | 42.4 |
| 3 | 63.1 | 58.5 | 45.2 | 47.1 | 64.0 | 59.2 | 46.1 | 47.8 |
| 4 | 58.9 | 53.7 | 50.6 | 49.7 | 62.8 | 57.8 | 54.5 | 53.8 |
| 5 | 57.4 | 52.8 | 54.5 | 52.1 | 66.4 | 62.2 | 63.4 | 61.5 |
| 6 | 58.9 | 55.1 | 56.8 | 54.3 | 71.7 | 68.6 | 69.6 | 67.8 |
| 7 | 58.0 | 54.8 | 55.4 | 53.1 | 75.9 | 74.0 | 73.2 | 72.3 |
| PTC-MR | | | | | | | | |
| 1 | 54.9 | 49.6 | 42.2 | 46.2 | 55.2 | 49.9 | 42.4 | 46.5 |
| 2 | 54.7 | 49.7 | 44.5 | 48.3 | 54.9 | 50.0 | 44.8 | 48.7 |
| 3 | 55.8 | 51.2 | 48.8 | 52.0 | 56.7 | 52.0 | 49.7 | 52.9 |
| 4 | 56.1 | 52.1 | 53.2 | 54.6 | 59.0 | 55.1 | 56.1 | 57.6 |
| 5 | 52.9 | 49.1 | 54.1 | 54.2 | 61.0 | 57.6 | 62.2 | 62.7 |
| 6 | 50.6 | 47.6 | 52.9 | 52.8 | 66.0 | 63.2 | 68.3 | 68.3 |
| 7 | 52.9 | 50.5 | 55.2 | 55.0 | 71.5 | 69.2 | 73.8 | 73.7 |

true positive rate) which corresponds to a single point in ROC space. Our ROC curve is created by connecting this point to (0,0) and (1,1) [6].

As it was mentioned earlier, subgraph isomorphism is a computationally challenging task. In our experiments we restricted the sizes of contrast graphs to eight (MUTAG) and seven (PTC) and performed a set of tests to present pattern size influence on classification results.

Table 2 shows cardinality of contrast and common sets of subgraphs for both datasets depending on maximal size of contrast graphs. Values were calculated as an average over all iterations performed during leave-one-out cross-validation procedure. Number of contrast subgraphs increases with maximal contrast subgraph size while number of common subgraphs remains constant.

Table 3, 4 show classification results for the examined datasets and maximal contrast subgraph sizes gained by CCPC classifier using different combinations of scoring routines. It can be observed that generally classification quality measures increase with maximal contrast subgraph size.

Results obtained with the scConB scoring scheme are more accurate than those obtained with scConA. According to Table 2 for the PTC datasets, the number of contrast graphs for negative class is always greater than that for the positive ones (for graph sizes greater than 1). In this case normalization of scores for both classes by dividing them by median score for each class improve classification results.

As far as common subgraphs scoring is concerned results obtained with scComB (using the growth rate) are better than scComA (using only support).

Table 5 shows comparison of accuracy and ROC score for different classifiers. We compare the results of our classifier (CCPC) with the following algorithms: SVM [17, 2]; Subdue [2]; SubdueCL [2]; Frequent Sub-Structure Based Approach [2]; Frequent Sub-Structure Based Approach with Sequential Rule Covering [2]. Our algorithm outperformed other approaches in both quality measures.

Table 5. Leave-one-out cross-validation results (accuracy and ROC score in %) for the MUTAG and PTC datasets using different algorithms. Best results for each quality measure are in bold face and second best are in italic face. *NA* indicates that certain value was not available.

| Algorithm | MUTAG | PTC-FM | PTC-FR | PTC-MM | PTC-MR |
|----------------------|-------------|-------------|-------------|-------------|-------------|
| Accuracy | | | | | |
| CCPC | 92.6 | 79.1 | 78.6 | 75.9 | 73.8 |
| SVM | <i>89.1</i> | <i>64.5</i> | <i>67.0</i> | <i>66.4</i> | <i>65.7</i> |
| ROC score | | | | | |
| CCPC | 90.1 | 76.8 | 72.8 | 74.0 | 73.7 |
| SVM | <i>NA</i> | 59.3 | 45.4 | 60.2 | 55.0 |
| Subdue | <i>NA</i> | 64.2 | 58.5 | 61.9 | 57.4 |
| SubdueCL | <i>NA</i> | 63.3 | 60.8 | 63.5 | 59.6 |
| Freq. Sub. | <i>NA</i> | 67.3 | 63.4 | 65.5 | 62.6 |
| Freq. Sub. Seq. Cov. | <i>NA</i> | <i>69.6</i> | <i>66.3</i> | <i>66.7</i> | <i>68.0</i> |

6 Conclusions

In this paper we presented a new approach for solving chemical compounds classification problem. Our algorithm (CCPC - Contrast Common Patterns Classifier) uses concepts of contrast and common subgraphs as well as some ideas characteristic for emerging patterns technique.

The results show that our algorithm outperformed the existing schemes. What is more, construction and structure of our classifier is simpler than approaches based on support vector machines (SVM). This feature lets the domain expert to modify the classifier with professional knowledge by adding new patterns to contrast or common subgraphs sets or by modifying their supports.

The main concept of our classifier is domain independent so it can be used to solve classification problems in other areas as well.

References

1. C. Borgelt and M. R. Berthold. Mining molecular fragments: Finding relevant substructures of molecules. In *ICDM '02: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)*, pages 51–58, Washington, DC, USA, 2002. IEEE Computer Society.
2. Mukund Deshpande, Michihiro Kuramochi, and George Karypis. Frequent substructure-based approaches for classifying chemical compounds. In *ICDM '03: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'03)*, pages 35–42, 2003.
3. R. Diestel. *Graph Theory*. Springer-Verlag, New York, USA, 2000.
4. G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. In *Knowledge Discovery and Data Mining*, pages 43–52, 1999.
5. Guozhu Dong, Xiuzhen Zhang, Limsoon Wong, and Jinyan Li. CAEP: Classification by aggregating emerging patterns. In *Discovery Science*, pages 30–42, 1999.
6. Tom Fawcett. Using rule sets to maximize ROC performance. In *ICDM*, pages 131–138, 2001.
7. S. Fortin. The graph isomorphism problem. Technical report, University of Alberta, Edmonton, Alberta, Canada, 1996.
8. J. Gonzalez, L. Holder, and D. Cook. Application of graph based concept learning to the predictive toxicology domain. In *PTC Workshop at the 5th PKDD*, 2001.
9. A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Principles of Data Mining and Knowledge Discovery*, pages 13–23, 2000.
10. H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *ICML: Proceedings of the 20th International Conference on Machine Learning*, pages 321–328, Washington, DC, USA, 2003.
11. R. Kotagiri and J. Bailey. Discovery of emerging patterns and their use in classification. In *AI 2003: Advances in Artificial Intelligence (LNCS 2903)*, pages 1–12, Perth, Australia, 2003.
12. M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM'01)*, pages 313–320, 2001.

13. F. J. Provost and T. Fawcett. Robust classification systems for imprecise environments. In *AAAI/IAAI*, pages 706–713, 1998.
14. R. C. Read and D. G. Corneil. The graph isomorph disease. *Journal of Graph Theory*, page 339363, 1977.
15. A. Srinivasan, R. D. King, S. H. Muggleton, and M. Sternberg. The predictive toxicology evaluation challenge. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 1–6. Morgan-Kaufmann, 1997.
16. A. Srinivasan, S. Muggleton, M. J. E. Sternberg, and R. D. King. Theories for mutagenicity: A study in first-order and feature-based induction. *Artificial Intelligence*, 85(1-2):277–299, 1996.
17. S. J. Swamidass, J. Chen, J. Bruand, P. Phung, L. Ralaivola, and P. Baldi. Kernels for small molecules and the prediction of mutagenicity, toxicity and anti-cancer activity. *Bioinformatics*, 21(1):359–368, 2005.
18. R. M. H. Ting and J. Bailey. Mining minimal contrast subgraph patterns. In *SIAM '06: Proceedings of the 2006 SIAM Conference on Data Mining*, Maryland, USA, 2006.
19. J. R. Ullmann. An algorithm for subgraph isomorphism. *J. ACM*, 23(1):31–42, 1976.
20. X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning*, 2002.

Power Transients Characterization and Classification Using Higher-Order Cumulants and Competitive Layers

Juan-José González de-la-Rosa^{1,3}, Antonio Moreno Muñoz^{2,3}, Isidro Lloret³,
Carlos G. Puntonet⁴, and Juan-Manuel Górriz⁴

¹ University of Cádiz, Electronics Area, EPSA, Av. Ramón Puyol S/N. E-11202, Algeciras-Cádiz, Spain

juanjose.delarosa@uca.es

² University of Córdoba, Electronics Area
Escuela Pol. Superior, Campus Rabanales, Córdoba, Spain

amoreno@uco.es

³ Research Group TIC168-Computational Instrumentation and Industrial Electronics

⁴ University of Granada, Dept. of Architecture and Computers Technology, ESII,
C/Periodista Daniel Saucedo. 18071, Granada, Spain

carlos@atc.ugr.es, gorriz@ugr.es

Abstract. This paper deals with power-quality (PQ) event detection, classification and characterization using higher-order sliding cumulants to examine the signals. Their maxima and minima are the main features, and the classification strategy is based in competitive layers. Concretely, we concentrate on the task of differentiating two types of transients (short duration and long duration). By measuring the fourth-order central cumulants' maxima and minima, we build the two-dimensional feature measured vector. Cumulants are calculated over high-pass digitally filtered signals, to avoid the low-frequency 50-Hz signal. We have observed that the minima and maxima measurements produce clusters in the feature space for 4th-order cumulants; 3rd-order cumulants are not capable of differentiate these two very similar PQ events. The experience aims to set the foundations of an automatic procedure for PQ event detection.

1 Introduction

Power quality event detection and classification is gaining importance due to worldwide use of delicate electronic devices. Things like lightning, large switching loads, non-linear load stresses, inadequate or incorrect wiring and grounding or accidents involving electric lines, can create problems to sensitive equipment, if it is designed to operate within narrow voltage limits, or if it does not incorporate the capability of filtering fluctuations in the electrical supply [1,2].

The solution for a PQ problem implies the acquisition and monitoring of long data records from the energy distribution system, along with an automated detection and classification strategy which allows to identify the cause of these

voltage anomalies. Signal processing tools have been widely used for this purpose, and are mainly based in spectral analysis and wavelet transforms. These second-order methods, the most familiar to the scientific community, are based on the independence of the spectral components and evolution of the spectrum in the time domain. Another tools are thresholding, linear classifiers and Bayesian networks. The goal of the signal processing analysis is to get a feature vector from the data record under study, which constitute the input to the computational intelligence modulus, which has the task of classification.

Some recent works bring a different strategy, based in higher-order statistics (HOS), in dealing with the analysis of transients within PQ analysis [2] and other fields of Science [3]. Without perturbation, the 50-Hz of the voltage waveform exhibits a Gaussian behavior. Deviations from Gaussianity can be detected and characterized via HOS. Non-Gaussian processes need third and fourth order statistical characterization in order to be recognized. In order words, second order moments and cumulants could be not capable of differentiate non-Gaussian events.

The situation described matches the problem of differentiating between a transient of long duration named fault (within a signal period), and a short duration transient (25 per cent of a cycle). This one could also bring the 50-Hz voltage to zero instantly and, generally affects the sinusoid dramatically. By the contrary, the long-duration transient could be considered as a modulating signal (the 50-Hz signal is the carrier). These transients are intrinsically non-stationary, so it is necessary a battery a observations (sample registers) to obtain a reliable characterization.

The main contribution of this work consists of the application of higher-order central cumulants to characterize PQ events (could be see as a complement of [2]), along with the use of a competitive layer as the classification tool. Results reveal that two different clusters, associated to both types of transients, can be recognized in the 2D graph. The successful results convey the idea that the physical underlying processes associated to the analyzed transients, generate different types of deviations from the typical effects that the noise cause in the 50-Hz sinusoid voltage waveform.

The paper is organized as follows: Sect. 2 summarizes the main equations of the cumulants used in the paper. Sect. 3 recalls the competitive layer's foundations, along with the *Kohonen* learning rule. The experience is described in Sect. 4, and the conclusions are drawn in Sect. 5.

2 Higher-Order Cumulants

High-order statistics, known as cumulants, are used to infer new properties about the data of non-Gaussian processes [4,5,6].

The relationship among the cumulant of r stochastic signals, $\{x_i\}_{i \in [1,r]}$, and their moments of order p , $p \leq r$, can be calculated by using the *Leonov-Shiryayev* formula [4,5,7,8]

$$\begin{aligned}
 Cum(x_1, \dots, x_r) &= \sum (-1)^{p-1} \cdot (p-1)! \cdot E\left\{ \prod_{i \in s_1} x_i \right\} \\
 &\quad \cdot E\left\{ \prod_{i \in s_2} x_j \right\} \cdots E\left\{ \prod_{i \in s_p} x_k \right\}
 \end{aligned} \tag{1}$$

where the addition operator is extended over all the partitions, like one of the form (s_1, s_2, \dots, s_p) , $p = 1, 2, \dots, r$; and $(1 \leq i \leq p \leq r)$; s_i is a set belonging to a partition of order p , of the set of integers $1, \dots, r$.

Let $\{x(t)\}$ be an r th-order stationary random process. The r th-order cumulant is defined as the joint r th-order cumulant of the random variables $x(t), x(t+\tau_1), \dots, x(t+\tau_{r-1})$,

$$\begin{aligned}
 C_{r,x}(\tau_1, \tau_2, \dots, \tau_{r-1}) \\
 = Cum[x(t), x(t+\tau_1), \dots, x(t+\tau_{r-1})].
 \end{aligned} \tag{2}$$

Considering $\tau_1 = \tau_2 = \tau_3 = 0$ in Eq. (2), we have some particular cases:

$$\gamma_{2,x} = E\{x^2(t)\} = C_{2,x}(0) \tag{3a}$$

$$\gamma_{3,x} = E\{x^3(t)\} = C_{3,x}(0, 0) \tag{3b}$$

$$\gamma_{4,x} = E\{x^4(t)\} - 3(\gamma_{2,x})^2 = C_{4,x}(0, 0, 0) \tag{3c}$$

Eqs. (3) are measurements of the variance, skewness and kurtosis of the distribution in terms of cumulants at zero lags (the central cumulants). Normalized kurtosis and skewness are defined as $\gamma_{4,x}/(\gamma_{2,x})^2$ and $\gamma_{3,x}/(\gamma_{2,x})^{3/2}$, respectively. We will use and refer to normalized quantities because they are shift and scale invariant.

3 Competitive Layers: A Brief Summary

The neurons in a competitive layer distribute themselves to recognize frequently presented input vectors. The competitive transfer function accepts a net input vector \mathbf{p} for a layer (each neuron competes to respond to \mathbf{p}) and returns neuron outputs of 0 for all neurons except for the winner, the one associated with the most positive element of net input. If all biases are 0, then the neuron whose weight vector is closest to the input vector has the least negative net input and, therefore, wins the competition to output a 1.

The winning neuron will move closer to the input, after this has been presented. The weights of the winning neuron are adjusted with the *Kohonen* learning rule. Supposing that the i th-neuron wins, the elements of the i th-row of the input weight matrix (\mathbf{IW}) are adjusted as shown in Eq. (4):

$$\mathbf{IW}_i^{1,1}(q) = \mathbf{IW}_i^{1,1}(q-1) + \alpha [\mathbf{p}(q) - \mathbf{IW}_i^{1,1}(q-1)], \tag{4}$$

where \mathbf{p} is the input vector, q is the time instant, and α is the learning rate parameter. The *Kohonen* rule allows the weights of a neuron to learn an input vector, so it is useful in recognition applications. Thus, the neuron whose weight vector was closest to the input vector is updated to be even closer. The result is that the winning neuron is more likely to win the competition the next time a similar vector is presented. As more and more inputs are presented, each neuron in the layer closest to a group of input vectors soon adjusts its weight vector toward those inputs. Eventually, if there are enough neurons, every cluster of similar input vectors will have a neuron that outputs 1 when a vector in the cluster is presented, while outputting a 0 at all other times. Thus, the competitive network learns to categorize the input vectors it sees.

4 Experimental Results

The aim is to differentiate between two classes of transients (PQ events), named long-duration and short-duration. The experiment comprises two stages. The feature extraction (classification) stage is based on the computation of cumulants. Each vector's coordinate corresponds to the local maximum and minimum of the 4th-order central cumulant. And the classification stage is based on the application of the competitive layer to the feature vectors, in order to obtain two clusters in the feature plane. We use a two-neuron competitive layer, which receives two-dimensional input feature vectors in this training stage.

We analyze a number of 16 1000-point (roughly) real-life registers during the feature extraction stage. Before the computation of the cumulants, two pre-processing actions have been performed over the sample signals. First, they have been normalized because they exhibit very different-in-magnitude voltage levels. Secondly, a high-pass digital filter (5th-order Butterworth model with a characteristic frequency of 150 Hz) eliminates the low frequency components which are not the targets of the experiment. This by the way increases the non-Gaussian characteristics of the signals, which in fact are reflected in the higher order cumulants.

After filtering, a 50-point sliding battery of central cumulants (2nd, 3rd and 4th order) are calculated. The window's width (50 points) has been selected neither to be so long to cover the whole signal nor to be very short. The algorithm calculates the 3 central cumulants over 500 points, and then it jumps to the following starting point; as a consequence we have 98 per cent overlapping sliding windows ($49/50=0.98$). Thus, each computation over a window (called a segment) outputs 3 cumulants.

Fig. 1 and Fig. 2 show an example of signal processing analysis of two sample registers corresponding to a long-duration and a short-duration events, respectively.

The 2nd-order cumulant sequence corresponds to the variance, which clearly indicates the presence of an event. Both types of transients exhibit an increasing variance in the neighborhood of the PQ event, that presents the same shape, with only one maximum. The magnitude of this maximum is by the way the

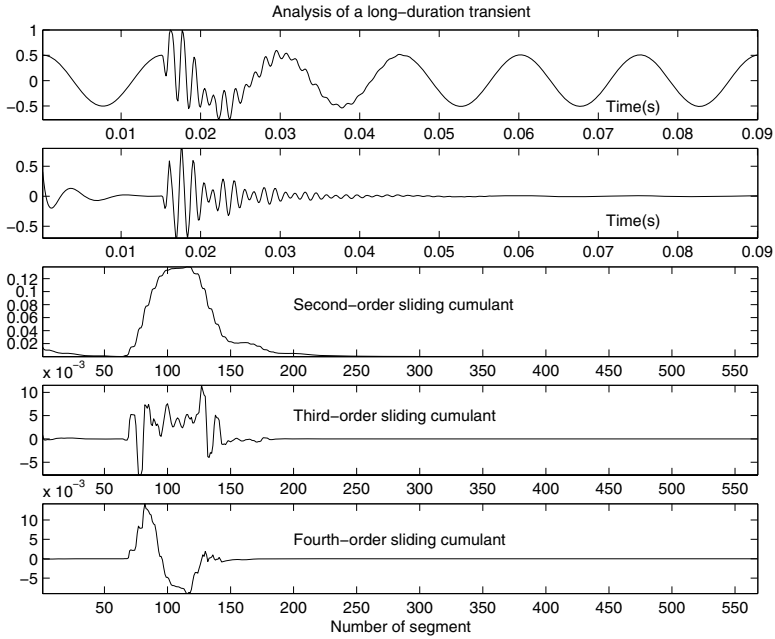


Fig. 1. Long duration transient analysis. From top to bottom: the original data record, the filtered sequence, 2nd-3rd-4th-order central cumulants sliding windows, respectively.

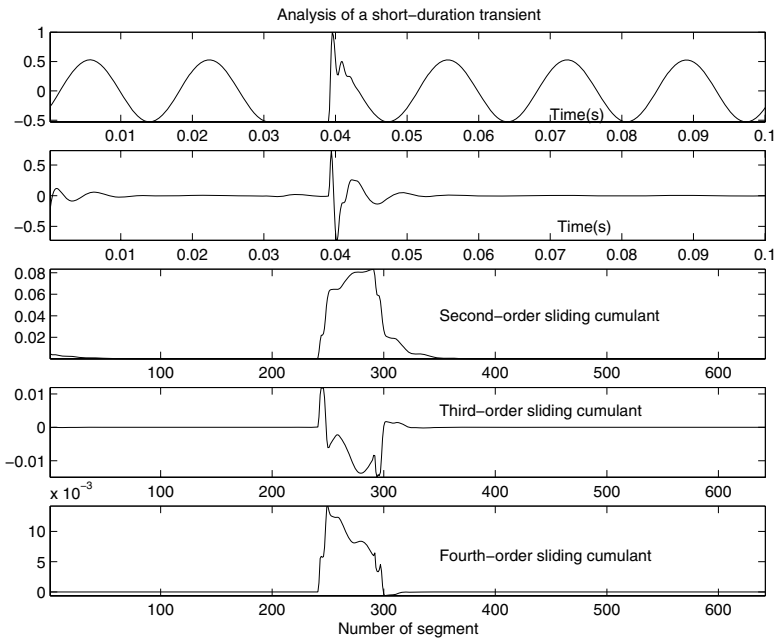


Fig. 2. Short duration transient analysis

only available feature which can be used to distinguish different events from the second order point of view. This may suggest the use of additional features in order to distinguish different types of events.

For this reason the higher-order central cumulants are calculated. An unbiased estimator of the cumulants has been selected. Third-order diagrams don't show quite different clusters if we consider a bi-dimensional space (2 coordinates for each feature vector) because maxima and minima are similar. It is possible to differentiate PQ events from the 3rd-order perspective if we consider more features in the input vector, like the number of extremes (maxima and minima), and the order in which the maxima and the minima appear as time increases. In this paper we have focussed the experience on a bi-dimensional representation (2-dimensional feature vectors) because we obtain very intelligible 2-D graphs.

Fourth-order sliding cumulants exhibit clear differences, not only for the shape of the computation graph (the bottom graph in Figs. 2 and 1), but also for the different location of minima, which suggest a clustering for the points.

Fig. 3 presents the results of the training stage, using the *Kohonen* rule. The horizontal (vertical) axis corresponds to the maxima (minima) value. Each cross in the diagram corresponds to an input vector and the circles indicate the final location of the weight vector (after learning) for the two neurons of the competitive layer. Both weight vectors point to the asterisk, which is the initializing point (the midpoint of the input intervals).

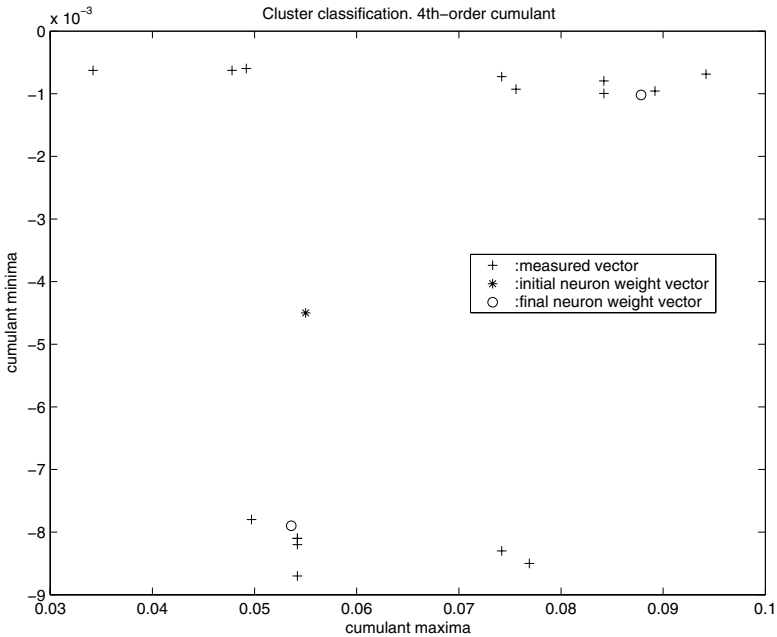


Fig. 3. Competitive layer training results over 20 epochs

The separation between classes (inter-class distance) is well defined. Both types of PQ events are horizontally clustered. The correct configuration of the clusters is corroborated during the simulation of the neural network, in which we have obtained an approximate classification accuracy of 97 percent. During the simulation new signals (randomly selected from our data base) were processed using the method described.

The accuracy of the classification method increases with the number of data. To evaluate the confidence of the statistics a significance test have been conducted. This informs if the number of experiments is statistically significant according to the fitness test [2]. As a result of the test, the number of measurements is significantly correct.

5 Conclusion

In this paper we have proposed a method to detect and classify two PQ transients, named short and long-duration. The method comprises two stages. The first includes pre-processing (normalizing and filtering) and outputs the 2-D feature vectors, each of which coordinate corresponds to the maximum and minimum of the central cumulants. The second stage uses a neural network to classify the signals into two clusters. This stage is different-in-nature from the one used in [2] consisting of quadratic classifiers. The configuration of the clusters is assessed during the simulation of the neural network, in which we have obtained an acceptable classification accuracy.

Acknowledgments. The authors would like to thank the *Spanish Ministry of Education and Science* for funding the project DPI2003-00878 which involves noise processes modeling, and the PETRI project PTR95-0824-OP involving higher-order statistics. Also thanks to the *Andalusian Government* for the trust put in the research group PAI-TIC-168, and for supporting the excellence project TIC-155, which involves higher-order statistics.

References

1. Moreno, A., Pallarés, V., De la Rosa, J.J.G., Galisteo, P.: Study of voltage sag in a highly automated plant. (In: MELECON 2006, Proceedings of the 2006 13th IEEE Mediterranean Electrotechnical Conference)
2. Ömer Nezih Gerek, Ece, D.G.: Power-quality event analysis using higher order cumulants and quadratic classifiers. *IEEE Transactions on Power Delivery* **21** (2006) 883–889
3. De la Rosa, J.J.G., Puntonet, C.G., Lloret, I., Górriz, J.M.: Wavelets and wavelet packets applied to termite detection. *Lecture Notes in Computer Science (LNCS)* **3514** (2005) 900–907 Computational Science - ICCS 2005: 5th International Conference, GA Atlanta, USA, May 22–25, 2005, Proceedings, Part I.
4. Nykias, C.L., Mendel, J.M.: Signal processing with higher-order spectra. *IEEE Signal Processing Magazine* (1993) 10–37

5. Mendel, J.M.: Tutorial on higher-order statistics (spectra) in signal processing and system theory: Theoretical results and some applications. *Proceedings of the IEEE* **79** (1991) 278–305
6. Nandi, A.K.: *Blind Estimation using Higher-Order Statistics*. 1 edn. Volume 1. Kluwer Academic Publishers, Boston (1999)
7. De la Rosa, J.J.G., Puntonet, C.G., Lloret, I.: An application of the independent component analysis to monitor acoustic emission signals generated by termite activity in wood. *Measurement* (Ed. Elsevier) **37** (2005) 63–76 Available online 12 October 2004.
8. Nikias, C.L., Petropulu, A.P.: *Higher-Order Spectra Analysis. A Non-Linear Signal Processing Framework*. Englewood Cliffs, NJ, Prentice-Hall (1993)

Mutual Information Estimation in Higher Dimensions: A Speed-Up of a k -Nearest Neighbor Based Estimator*

Martin Vejmelka¹ and Kateřina Hlaváčková-Schindler²

¹ Institute of Computer Science, Academy of Sciences of the Czech Republic
Pod Vodárenskou Věží 2, 18207 Praha 8, Czech Republic
vejmelka@cs.cas.cz

² Commission for Scientific Visualization, Austrian Academy of Sciences
Donau-City Str. 1, A-1220 Vienna, Austria
katerina.schindler@assoc.oew.ac.at

Abstract. We focus on the recently introduced nearest neighbor based entropy estimator from Kraskov, Stögbauer and Grassberger (KSG) [10], the nearest neighbor search of which is performed by the so called box assisted algorithm [7]. We compare the performance of KSG with respect to three spatial indexing methods: box-assisted, k -D trie and projection method, on a problem of mutual information estimation of a variety of pdfs and dimensionalities. We conclude that the k -D trie method is significantly faster than box-assisted search in fixed-mass and fixed-radius neighborhood searches in higher dimensions. The projection method is much slower than both alternatives and not recommended for practical use.

1 Introduction

Shannon **differential entropy** $H(S)$ of random variable X with a continuous density $\mu(x)$ is defined as (Shannon, [16]):

$$H(X) = \int_{-\infty}^{\infty} \mu(x) \log(x) dx, \quad (1)$$

where \log is natural logarithm. The **mutual information** $I(X, Y)$ between two random variables X with marginal pdf $p_X(x)$ and Y with marginal pdf $p_Y(y)$ is [16])

$$I(X, Y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \mu(x, y) \log \frac{\mu(x, y)}{p_X(x)p_Y(y)}. \quad (2)$$

and can be computed as

* The first author was supported by the 6RP EU project BRACCIA (Contract No 517133 NEST). The second author was supported by the grant of Austrian Research Fonds FWF-H-226 (2005) under Charlotte Bühler Program and by ASCR IET 100 750 401, Project Baddy.

$$I(X, Y) = H(X) + H(Y) - H(X, Y). \tag{3}$$

Since the introduction of Shannon entropy in 1948 [16], there have been a variety of parametric and non-parametric methods developed for entropy estimation (for a review of non-parametric estimators, see e.g. Beirlant [1]. Mutual information (MI) is important to many areas and is difficult to be estimated [13]. The most widespread method to approximate MI is the histogram method, which suffers from the curse of dimensionality and has systematic errors as well as Parzen window estimator [12]. Kozachenko and Leonenko developed a consistent entropy estimator (KL) based on nearest neighbor search [9], [6]. Kraskov et al. in [10] modified their algorithm into the so called KSG algorithm and improved its performance on a variety of pdfs, as well as its applicability in higher dimensional spaces. Compared to other methods in multivariate spaces (Van Hulle [18]), KSG showed a very good precision also for non-Gaussian pdfs (important e.g. for Independent Components Analysis [8]). To summarize, KSG seems to be currently the most promising entropy and mutual information estimator. The authors of KSG selected the box assisted algorithm ([15]) as the main search method. In this paper we focus on analyzing the box-assisted algorithm and recommending fast and simple alternatives for performing spatial queries in higher dimensions.

Section 2 presents the idea of the KSG method. The box-assisted algorithm is described in Section 3. Section 4 presents the k -D trie algorithm and Section 5 the projection method. The setup and results of our experiments are analysed in Section 6. Section 7 summarizes our results.

2 Estimators of Entropy and Mutual Information Using Nearest Neighbor Search

First we shortly mention the entropy estimator KL introduced by Kozachenko and Leonenko in [9], necessary for understanding the KSG method. For a detailed derivation of both methods see [10] or [11]. For reasons of simplicity, we consider the KL estimator in R^2 . The idea is, for each point $z_i = (x_i, y_i) \in R^2$ to rank its neighbors by distance $d_{i,j} = \|z_i - z_j\| : d_{i,j_1} \leq d_{i,j_2} \leq \dots$, supposing $\|\cdot\|$ be a metrics) and then to estimate $H(X)$ from the average distance to the k -nearest neighbor, averaged over all x_i . Shannon entropy (1) can be understood as an average of $\log \mu(x)$ and then $\log \mu(x_i) \approx \psi(k) - \psi(N) - mE(\log \epsilon) - \log c_m$, which leads to the KL entropy estimator

$$\hat{H}(X) = -\psi(k) + \psi(N) + \log(c_m) + \frac{m}{N} \sum_{i=1}^N \log \epsilon(i). \tag{4}$$

Let $Z = (X, Y)$ be a random variable. The estimator of $H(Z)$ differs from (4) in that x is replaced by z , d is replaced by $d_Z = d_X + d_Y$ and c_d by $c_{d_X} c_{d_Y}$:

$$\hat{H}(X, Y) = -\psi(k) + \psi(N) + \log c_{d_X} c_{d_Y} - ((d_X + d_Y)/N) \sum_{i=1}^N \log \epsilon(i). \tag{5}$$

Formula (3) can be applied for the same k . This would however mean that different distance scales would be effectively used in the joint and marginal spaces. However, the biases of formula (4) resulting from the nonuniformity of the density in marginal spaces would be different for the estimates $H(X)$, $H(Y)$ and $H(X, Y)$ and would not cancel. To avoid this, Kraskov et al. recommend not to use fixed k for marginal entropy estimation. Two estimators are proposed for dimension m , but here we consider only the first one (for simplicity reasons). Assume that the k -th neighbor of x_i is on one of the vertical sides of the square of size $\epsilon(i)$ (in two dimensions). Then if there are altogether $n_x(i)$ points within the vertical lines $(x_i - \epsilon(i)/2, x_i + \epsilon(i)/2)$, then $\epsilon(i)/2$ is the distance to the $(n_x(i) + 1)$ -th neighbor of and x_i and the KSG entropy estimator is

$$\hat{H}(X) = -\frac{1}{N} \sum_{i=1}^N \psi[n_x(i) + 1] + \psi(N) + \log c_{d_X} + \frac{d_X}{N} \sum_{i=1}^N \log \epsilon(i). \quad (6)$$

The first m -dimensional MI KSG estimator in [10] can be written as:

$$I^{(1)}(X_1, \dots, X_m) = \psi(k) - (m - 1)\psi(N) - \langle \psi(n_{x_1}) + \dots + \psi(n_{x_m}) \rangle, \quad (7)$$

where $\langle \dots \rangle = (1/N) \sum_{i=1}^N E[\dots(i)]$ and n_{x_i} is the number of points x_j so that $\|x_j - x_i\| < \epsilon(i)/2$. More details, including some hints for selection of parameter k , influencing the precision of approximation, can be found in [10]. According to the authors, the above estimator has a very small bias. The KSG algorithm spends most of its time in spatial queries. The steps necessary for the computation of a single point contribution to the MI estimate are the following:

1. In the space Z , find the distance to the k -th nearest neighbor $\epsilon(k)$ from the given reference point (*fixed-mass-search*),
2. In the spaces X and Y , find out how many points are in the hypersphere of radius $\epsilon(k)$ centered around the projection of a reference point to the respective subspace (*fixed-radius-search*).

Let us note here, that the fixed-mass-query is always performed in the space Z , having the highest dimensionality; this query will take most of the computation time. It is of utmost importance to find the best spatial indexing strategy for all three cases. In this paper, we examine possible algorithms and give conditions under which it is advantageous to apply a given spatial index. We base our analysis on the maximum (L_∞) norm. This is however not a requirement of the KSG algorithm itself.

3 The Box Assisted Algorithm

Here we describe the box-assisted algorithm as presented by Schreiber in [15], developed by Grassberger in [7] and associated with the KSG algorithm in [10] and [11]. The basic idea of the box spatial indexing structure is to select two dimensions and cover the projection space with equally sized boxes (squares).

Any point of the observation space can be projected into exactly one of the boxes. All of the points can be sorted into the boxes in $O(N)$ time. It is possible to use an array to store the sets of points contained in each box as a linked list of indices. Schreiber [15] recommends using pointers but this is computationally very expensive unless advanced allocation techniques are employed. When a fixed range search is required, first the box containing the reference point is found; secondly, the boxes that must be searched around can be efficiently identified from the query radius $\epsilon > 0$. If the query radius is known, then the boxes should have the size of the query radius. In this way, only 9 boxes need to be searched around a reference point regardless of the dimensionality of the observation space. If a k -nearest neighbor search is required then it is necessary to proceed from the box with the reference point along a (two dimensional) Peano curve [14] (generally a 'spiral' entirely filling an m -dimensional space).

The time complexity of the accelerated box-assisted algorithm (see [15]) is $O(N \log N)$.

4 k -D Trie

This method is a recursive space-partitioning data structure for organizing points in an k -dimensional space proposed by Friedman et al. [5]. A k -D trie is a variation on the k -D tree that only stores points in the leaf nodes, sometimes in small collections called bins or buckets. Technically, the k in k -D trie refers to the number of dimensions; however the designation k -D trie is usually used whatever the dimension of the observation space. In the following we will use k to refer to the number of neighbors to be consistent with the authors of the KSG algorithm.

A k -D trie is a structure that partitions the m -dimensional observation space using $m - 1$ dimensional hyperplanes. The simplest procedure of constructing a k -D trie has only one parameter *min-points-before-split* and can be described as follows, for details please refer to [5].

The algorithm begins with gathering the points and sorting them along their first coordinate. Then the median is found and its first coordinate is stored as the pivot coordinate. The set is split into two subsets - one smaller than the pivot and the second one larger. Tries can be broken arbitrarily and the pivot may be assigned to either one. The above procedure is repeated on each of the subsets while cycling through the coordinates of the points. If at any time a subset has a smaller number of elements than *min-points-before-split*, the set is not split anymore, but is stored as belonging to the current node and the node is denoted as a leaf node. The result of this construction procedure is a tree which stores groups of data points in each of its leaf nodes. The construction procedure can be improved at the costs of programming and run-time complexity. Here we will work only with the basic procedure described above. The cost of the above tree construction has complexity $O(N \log N)$. The search procedure for k -nearest neighbors briefly follows (for a detailed reference, please refer to [5]): Basically, it is necessary to find the leaf node of the tree which contains the reference point.

One searches all the points in that node and recurses back up the descended path while searching other subtrees if it is determined that they could contain some closer points. Fixed range search is performed similarly but the criterion for descending into subtrees is different. The test condition is whether the region to be searched overlaps with the part of the observation space partitioned by that subtree. Both of the above tests can be implemented efficiently.

5 Projection Method

The idea of the projection method (Friedman et al., [4]) is to find the best projection axis in R^m minimizing the number of distance computations. In other words, for a given test point, to find the axis with the smallest density of prototypes around the test point. The algorithm can be applied with any Minkowski metrics for $1 \leq p \leq \infty$.

Basic Procedure. The preprocessing for this algorithm consists basically of ordering the data points on the values of one of the coordinates. For each test point, the data points are examined in the order of their projected distance from the test point on the sorted coordinate. When this projected distance becomes larger than the distance (in the full dimensionality) to the k closest point of those data points already examined, no more data need to be considered. The k closest points of the examined points are those for the complete set.

Full Procedure. The points are ordered on several or all of the coordinates and the one with the smallest local projected density in the neighborhood of the test point is chosen. For each test point, the local projected sparsity on each axis is estimated as $s_i = |X_{i,p_i+\frac{q}{2}} - X_{i,p_i-\frac{q}{2}}|$, where $X_{i,j}$ is the i -th coordinate of the j -th ordered data point and p_i is the position of the test point in the i -th projection (q is the number of prototypes over which the sparsity is averaged on each projection). The prototype ordering on that particular coordinate, for which s_i is maximum, is chosen. Parameter q should correspond to a distance of about $2E[r_m]$. The computational time of the projection method is $O(k^{1/m}N^{1-1/m})$ and preprocessing has $O(N \log N)$ complexity. The storage of qN additional memory locations and for $q = m$, doubles the memory over that required by the naive method.

6 Experiments

The results of experiments of Schreiber ([15] comparing the box assisted algorithm to k-D trees for $m \leq 4$ (implementation from Bingham and Kot [3]) and to the naive algorithm were for fixed ϵ slightly better for box-assisted algorithm than for k-D trees (by factors at most 1.75 times). We obtained similar results for these dimensions and k-D trie; however when one extends the tests to the higher dimensions, the behavior of each method differs widely. In all cases, the size of the data set was 10000 samples, connected to the estimation of MI of

various pdf's. We focused on those sizes of $0 < \epsilon \ll 1$ for which the box assisted algorithm achieved the fastest speed. In our experiments, we compared the computational effectiveness of conducting k -nearest neighbor and fixed range queries for various pdfs with the box-assisted algorithm, k -D trie, projection method and naive search algorithm. We studied the dependence of the computational time on the neighborhood size (*fixed-radius-search*), as well as on the number of neighbors k (*fixed-mass-search*) in various space dimensions ($m = 2, 3, 8, 16, 32$). The considered pdfs were the uniform, $N(0, 1)$, exponential(1), Poisson(1) and the Gamma(2,2). All data points were scaled to fit into the unit hypercube in R^m . In the following we summarize the relative speed improvements of each method against the naive search.

6.1 Fixed Radius Search

In the fixed range tests the box-assisted algorithm tends to be better than the k -D trie for the dimensions m up to 4 (Fig. 1(a)); however the relative speed-up is small, the maximum is a factor of 2. For higher dimensions, the k -D trie is clearly better with a much larger relative speed-up, in tens and sometimes even hundreds. An exception is the uniform distribution, where the box-assisted search is still competitive in 8 dimensions for small ϵ neighborhoods. The box assisted search was designed with the uniform distribution in mind, so one can expect that the method works well with this pdf. Furthermore, uniform pdf does not require the algorithm to adapt to the data as the point density is similar in all parts of the space. The projection (Friedman) method often produces relative speed-ups in units, which is theoretically a good result, however the method is too slow to be applied in practice and inappropriate for large data sets such as the ones tested. For most pdfs, none of the methods produces any reasonable speed advantage over naive search in the 32-dimensional problems but sometimes a significant speed-up is achieved even in higher dimensions (Fig. 1(b)).

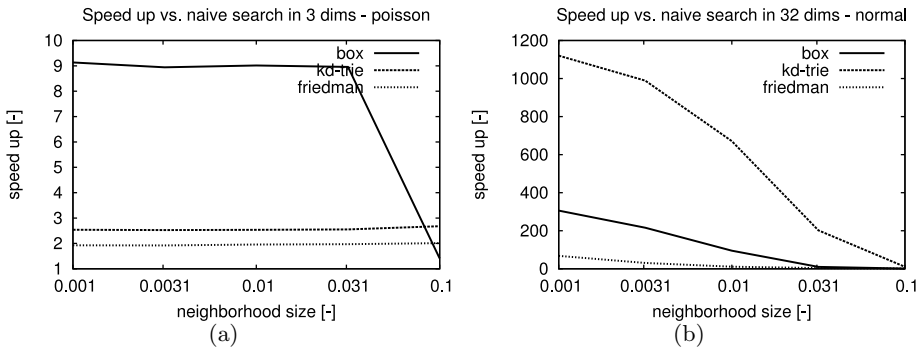


Fig. 1. Fixed radius neighborhood search relative speed-ups w.r.t. naive algorithm

6.2 Fixed Mass Search

When a fixed number of nearest neighbors is sought, the k -D trie method is significantly faster in almost all of the cases except for 2-dimensional problems; in this case, all the methods are comparable in efficiency for 1 or 2 neighbors. An exception to the rule is the uniform distribution where the box method is about 3 times faster than k -D trie in 2 dimensions for $k = 1$. However there are also 2 dimensional settings where the k -D trie technique can be one much faster than the box assisted search (Fig. 2(a)). For 32 dimensions, there almost no speed-up in any of the methods. The projection method is completely inappropriate for $k \geq 4$ because the repeated neighbor searches are very time-consuming. The improvement of the k -D trie method over the box assisted search is more significant when a higher number of neighbors is sought (Fig. 2(b)).

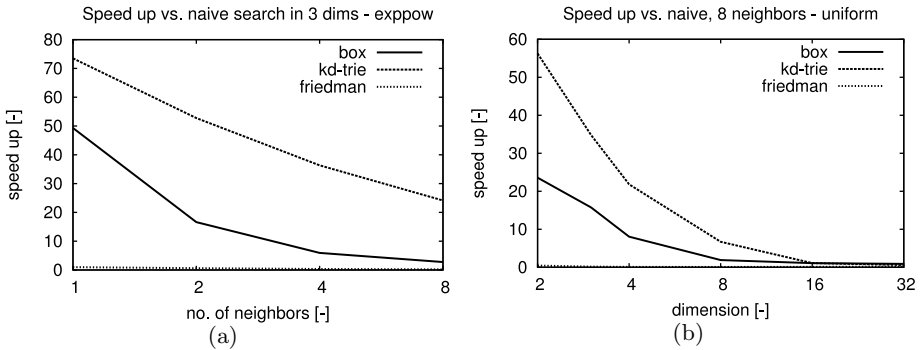


Fig. 2. Fixed mass neighborhood search relative speed-ups w.r.t. naive algorithm

7 Conclusion

In conclusion, we would like to evaluate the soundness of the motivations for selection of the box-assisted algorithm for the KSG method. Some improvements of this method have been published to date. For example, Theiler [17] reduced the speed of the box-assisted algorithm to $O(N)$ but only for dimensions $m \leq 3$ under assumption that the data set is not too singularly distributed (a criterion for this phenomenon was however not given). Schreiber [15] suggests a method for coarsening the grid size ϵ by a linear factor greater than 1. The overall conclusion in [15] was that the differences between the box assisted and the k D-tree algorithm are not pronounced enough to prefer one of the methods in general (k -D trie algorithm was not by Schreiber considered). We consider the basic version of the k -D trie algorithm to be very simple to code. Schreiber [15] states that the box-assisted algorithm is faster than the simple k D-tree method for fixed radius searches in data sets with lower dimensionality. Our numerical experiments indicate that when comparing box-assisted search against a k -D trie method, the last statement holds for the dimensions up to 3; the k -D trie

tends to be significantly more efficient for dimensions greater than 3. In higher dimensions, we recommend using the k -D trie method *both* for scanning fixed-mass neighborhoods and for scanning fixed-radius neighborhoods.

References

1. Beirlant, J., Dudewitz, E.J., Györfi, L., and van der Meulen, E.C.: Nonparametric entropy estimation: An overview. *Int. J. Math. And Statistical Sciences*, **6**, (1997) 17-39
2. Bentley, J.L.: Multidimensional binary search trees used in associative searching, *Communications of the ACM*, **18**, 9 (1975), 509-517
3. Bingham S., Kot M.: Multidimensional trees, range searching, and a correlation dimension algorithm of reduced complexity, *Phys Lett. A* **140**, 327 (1989)
4. Friedman, J.H., Baskett, F., Shustek, L.J.: An Algorithm for finding nearest neighbor. *IEEE Transactions on Computers* (1975) 1000-1006
5. Freidman, J.H., Bentley, J.L., and Finkel, R.A., An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.* **3**, 3 (Sep. 1977), 209-226
6. Goria, M.N., Leonenko, N.N., Mergel, V.V., Novi Inverardi, P.L., A new class of random vector entropy estimators and its applications in testing statistical hypotheses, *Nonparametric Statistics*, **17**, 3, 2005, 277-297
7. Grassberger, P.: An optimized box-assisted algorithm for fractal dimensions. *Phys. Lett.A*, **148** (1990) 63-68
8. Hyvaerinnen, A., Karhunen, J., Oja, E.: *Independent Component Analysis*, Wiley 2001
9. Kozachenko, L.F., Leonenko, N.N.: Sample estimate of the entropy of a random vector. *Problems of Information Transmission*, **23**(2) (1987) 95101
10. Kraskov, A., Stögbauer, H., Grassberger, P: Estimating mutual information, *Physical Review E* **69**, 066138 (2004)
11. Kraskov, A.: Synchronization and interdependence measures and their applications to the electroencephalogram of epilepsy patients and clustering of data, PhD thesis, John von Neumann Institute for Computing, 2004
12. Kwak, N., Choi, CH.: Input feature selection by mutual information based on Parzen window. *IEEE Trans. On Pattern Analysis and Machine Intellingence*, **24** (12) (2002) 1667-1671
13. Paninski, L: Estimation of entropy and mutual information. *Neural Computation* **15** (2003) 1191-1253
14. <http://en.wikipedia.org/wiki/Space-filling-curve>
15. Schreiber, T: Efficient neighbor searching in nonlinear time series analysis. In: *Int. Journal of Bifurc. and Chaos* **5**(2) (1995) 349-358
16. Shannon, C.: A mathematical theory of communication. *Bell System Tech. J.*, **27** (1948) 379-423
17. Theiler, J.: Efficient algorithm for estimation the correlation dimension from a set of discrete points, *Physical Review A* **36** (1987) 44564462
18. Van Hulle, M.M.: Edgeworth approximation of multivariate differential entropy, *Neural Computation* **17** (2005) 1903-1910

Grammar-Based Classifier System for Recognition of Promoter Regions

Olgierd Unold

The Institute of Computer Engineering, Control and Robotics
Wroclaw University of Technology
Wyb. Wyspianskiego 27, 50-370 Wroclaw, Poland
olgierd.unold@pwr.wroc.pl
<http://sprocket.ict.pwr.wroc.pl/~unold>

Abstract. Identifying bacterial promoters is an important step towards understanding gene regulation. In this paper, we address the problem of predicting the location of promoters in *Escherichia coli*. Language of bacterial sequence can be described using formal system such a context-free grammar, and problem of promoter region recognition replaced by grammar induction. The accepted method for this problem is to use grammar-based classifier system (GCS).

1 Introduction

Since a biological sequence is usually represented as a text that consists of a finite set of characters that represent nucleotides or amino acids, designing models based on formal languages have been constantly proposed since the early era of bioinformatics. Formal biosequence linguistic research has used finite-state automata, stochastic grammars based on hidden Markov models [5], and grammars based on computational logic [27-30]. The logic grammar approach to DNA language analysis involved mainly representing structures of a biological sequence in Definite Clause Grammar (DCG) and Prolog [23], [3], [25] or in systems of equivalent representational power to DCGs [16]. Formulation of DNA patterns in any formal grammar requires support of human (time consuming, as well as error prone method) and/or machine learning methods, such as knowledge-based neural network [31], [34], [16] or grammatical inference methods [26]. It is worth mention that the last approach concentrated mainly on the estimation of probability parameters of stochastic grammars while the problem of learning the structure of grammars remains a difficult task with a few positive results on biological sequences.

In this paper a new grammatical inference approach to learning formal grammar for DNA sequence is proposed. It is also assumed that that language of biological sentences is expressed by (non stochastic) context-free grammar (CFG). CFG is induced by grammar-based classifier system (GCS), a new model of learning classifier system.

The use of GCS will be demonstrated in recognition of *Escherichia coli* promoter sequences, which are probably the most studied and cited sequences in molecular biology.

CFG induction problems are sketched in second paragraph. Third section contains description of learning classifier systems, GCS preceded by short introduction to context-free grammars is presented in fourth paragraph. Fifth section contains biological preliminaries and presents the testbeds. Sixth section shows experimental results in promoter region recognition, whereas seventh section is a short summary.

2 Context-Free Grammar Induction

The process in which a system produces a grammar given a set of corpora is known as grammatical inference or grammar induction [6]. In general, the corpora may contain both positive and negative examples from the language under study, which is described most often by CFG. There are very strong negative results for the learnability of CFG. The main theorems are that it is impossible to evolve suitable grammar (each of the four classes of languages in the Chomsky hierarchy) only from positive examples [6], and that even the ability to ask equivalence queries does not guarantee exact identification of context-free language in polynomial time [1]. Effective algorithms exist only for regular languages, thus construction of algorithms that learn context-free grammar is critical and still open problem of grammar induction [7]. The approaches taken have been to provide learning algorithms with more helpful information, such as negative examples or structural information; to formulate alternative representation of CFGs; to restrict attention to subclasses of context-free languages that do not contain all finite languages; and to use Bayesian methods [16]. Grammar induction can be considered as a difficult optimization task. Evolutionary approaches are probabilistic search techniques especially suited for search and optimization problems where the problem space is large and complex. Many researchers have attacked the problem of grammar induction by using evolutionary methods to evolve (stochastic) CFG or equivalent pushdown automata [41], [40], [9], [4], [19], [13], [18], [10], [32], [11], [12], [35], but mostly for artificial languages like brackets, and palindromes. Grammar-based classifier system can be considered as a new evolutionary model. For surveys of the non-evolutionary approaches for CFG induction see [18].

3 Learning Classifier Systems

An ordinary Learning Classifier System (LCS), proposed by Holland [8], consists of three main elements: production system, rule rewarding system, and genetic algorithm (GA) – the discovery component. LCS learns by interacting with an environment. Production system receives a message from input detectors and posts a new one using output effectors. All messages are stored in a message list. A finite population of “condition-action” rules – classifiers – are kept to represent the current state of the system’s knowledge. These rules are used to accept incoming message and generate the new ones. Messages have the form of the binary vectors. Each classifier consists of the action part – which is also in the form of the binary vector – and the condition part – which is the ternary vector with elements in $\{0, 1, \#\}$ where # is a “don’t care”

symbol. Classifier matches message if there are matching symbols in all vector positions. The “don’t care” symbol matches both 0 and 1 in the message vector.

In every production cycle each message from the message list is compared with the condition part of the classifiers. Matching rules are chosen to bid in the auction. The bids are proportional to the strength of the classifiers. Winner rule is “activated” which means that it posts its action to the message list. Rule rewarding system (also known as the reinforcement component) distributes the bids among the classifiers that were activated. The reward increases the classifiers strength.

The discovery component is triggered every number of production cycles. GA chooses randomly, with the probability proportional to their fitness, two classifiers from the population. It applies crossover and mutation generating two new classifiers.

For many years the research on LCS was done on Holland’s classifier system. All implementations shared more or less the same features which can be summarized as follows (i) some form of a bucket brigade algorithm was used to distribute the rewards, (ii) evolution was triggered by the strength parameters of classifiers, (iii) the internal message list was used to keep track of past input [14].

During the last years new models of Holland’s system have been developed. Among others, two models appear particularly worth mentioning. The XCS classifier system [39] uses Q-learning to distribute the reward to classifiers, instead of bucket brigade algorithm; the genetic algorithm acts in environmental niches instead of on the whole population; and most important, the fitness of classifiers is based on the accuracy of classifier predictions, instead of the prediction itself. Stolzmann’s ACS [33] differs greatly from other LCS models in that ACS learns not only how to perform a certain task, but also an internal model of the dynamics of the task. In ACS classifiers are not simple condition-action rules but they are extended by an effect part, which is used to anticipate the environmental state.

4 Grammar-Based Classifier System

The GCS was widely described in previous works [36-38]. The system operates similar to the classic LCS but differs from them in (i) representation of classifiers population, (ii) scheme of classifiers’ matching to the environmental state, (iii) methods of exploring new classifiers.

Population of classifiers has a form of a context-free grammar rule set in a Chomsky Normal Form. This is not a limitation actually because every CFG can be transformed into equivalent CNF. Chomsky Normal Form allows only production rules in the form of $A \rightarrow a$ or $A \rightarrow BC$, where A, B, C are the non-terminal symbols and a is a terminal symbol. The first rule is an instance of *terminal rewriting rule*. These ones are not affected by the GA, and are generated automatically as the system meets unknown (new) terminal symbol. Left hand side of the rule plays a role of classifier’s action while the right side a classifier’s condition. System evolves only one grammar according to the so-called Michigan approach. In this approach each individual classifier – or grammar rule in GCS – is subject of the genetic algorithm’s operations. All classifiers (rules) form a population of evolving individuals. In each cycle a fitness calculating algorithm evaluates a value (an adaptation) of each classifier and a discovery component operates only on a single classifier.

Automatic learning CFG in GCS is realized with so-called grammatical inference from text [6]. According to this technique system learns using a training set that in this case consists of sentences both syntactically correct and incorrect. Grammar which accepts correct sentences and rejects incorrect ones is able to classify unseen so far sentences from a test set. Cocke-Younger-Kasami (CYK) parser, which operates in $\Theta(n^3)$ time [42], is used to parse sentences from corpus.

Environment of classifier system is substituted by an array of CYK parser. Classifier system matches the rules according to the current environmental state (state of parsing) and generates an action (or set of actions in GCS) pushing the parsing process toward the complete derivation of the sentence analyzed.

The discovery component in GCS is extended in comparison with standard LCS. In some cases a “covering” procedure may occur, adding some useful rules to the system. It adds productions that allow continuing of parsing in the current state of the system. This feature utilizes for instance the fact that accepting 2-length sentences requires separate, designated rule in grammar in CNF.

Apart from the “covering” a GA also explores the space searching for new, better rules. First GCS implementation used a simple rule fitness calculation algorithm which appreciated the ones commonly used in correct recognitions. Later implementations introduced the “fertility” technique, which made the rule fitness dependant on the amount of the descendant rules (in the sentence derivation tree) [37]. In both techniques classifiers used in parsing positive examples gain highest fitness values, unused classifiers are placed in the middle while the classifiers that parse negative examples gain lowest possible fitness values.

GCS uses a mutation of GA that chooses two parents in each cycle to produce two offspring. The selection step uses the roulette wheel selection. After selection a classical crossover or mutation can occur. Offspring that are created replace existing classifiers based on their similarity using crowding technique, which preserves diversity in the population and extends preservation of the dependencies between rules by replacing classifiers by the similar ones.

5 Biological Preliminaries and Experimental Testbeds

During the last years many prokaryotic genomes have been sequenced, including that of *Escherichia coli* [2]. The gene content of these genomes was mostly computationally recognized. However, the promoter regions are still undetermined in most cases and the software able to accurately predict promoters in sequenced genomes is not yet available in public domain. Promoter recognition, the computational task of finding the promoter regions on a DNA sequence, is very important for defining the transcription units responsible for specific pathways (because gene prediction alone cannot provide the solution) and for analysis of gene regulation. A promoter enables the initiation of a gene expression after binding with an enzyme called RNA polymerase, which moves bidirectionally in searching for a promoter and starts making RNA according to the DNA sequence at the transcription initiation site following the promoter [20], [17]. The most significant patterns in *E.coli* promoter sequences are the -10 and -35 regions, which are approximately at the region of 10 bases and 35 bases before the transcription initiation site. The spacing

(gap) between the -10 and -35 regions is not fixed, ranging from 15 to 19 bases. The -35 and -10 sequences together are the contact region for RNA polymerase.

The genome is treated by GCS as a string composed of letters $\{A, C, T, G\}$. The goal is, given an arbitrary potential promoter region to be able to find out whether it is true or false promoter region. As the learning set the database contributed by M. Noordewier and J. Shavlik to UCI repository [21] was used. The database consists of 53 positive instances and 53 negative instances, 57 letters each. Negative learning sentences were derived from *E. coli* bacteriophage *T7* believed to not contain any promoter sites. In order to get an estimate of how well the algorithm learned the concept of promoter, the test set consisting of unseen 36 instances including 18 positive and 18 negative examples was prepared. Positive test instances were prepared by mutating the bases of the randomly chosen positive learning sentences in non-critical positions, negative test instances by mutating in any positions of randomly chosen negative learning sentences. This method increases the amount of available examples and was first proposed in [22].

6 The Experiments

Evolution on learning promoter database ran for 5,000 generations, with the following genetic parameters: number of nonterminal symbols 19, number of terminal symbols 4, crossover probability 0.2, mutation probability 0.8, population consisted of maximal 150 classifiers where 130 of them were created randomly in the first generation, crowding factor 18, crowding size 3. The experiment was repeated 10 times because GCS uses random classifiers during initialization and learning.

After each execution four numbers were calculated: True Positives (correctly recognized positive examples), True Negatives (correctly recognized negatives), False Negatives (positives recognized as negatives), and False Positives (negatives recognized as positives). Then the average of these numbers were found and the following measures were calculated: Specificity, Sensitivity, and Accuracy. Specificity is a measure of the incidence of negative results in testing all the non-promoter sequences, i.e. $(\text{True Negatives}/(\text{False Positives} + \text{True Negatives})) \times 100$. Sensitivity is a measure of the incidence of positive results in testing all the promoter sequences, i.e. $(\text{True Positives}/(\text{True Positives} + \text{False Negatives})) \times 100$. Accuracy is measured by the number of correct results, the sum of true positives and true negatives, in relation to the number of tests carried out, i.e. $((\text{True Positives} + \text{True Negatives})/\text{Total}) \times 100$. GCS achieved 74.5% accuracy, 87.5% specificity, and 62.5% sensitivity in the learning set. Much more interesting are the results gained during generalization tests on the previously unseen examples from test set. Table 1 compares the results of GCS and two formal system based methods presented in [16].

Table 1. Different promoter recognition methods compared

| Method | Specificity | Sensitivity | Accuracy |
|--------|-------------|-------------|----------|
| KBANN | 97 | 16 | 56 |
| WANN | 82 | 69 | 75 |
| GCS | 94 | 61 | 78 |

Leung et al. [16] introduced Basic Gene Grammars (BGG) to represent many formulations of the knowledge of *E.coli* promoters. BGG is able to represent knowledge acquired from knowledge-based artificial neural network learning (KBANN approach [34]), and combination of grammar of weight matrices [24] and KBANN (denoted as WANN). Development of BGG is supported by DNA-ChartParser. The method was tested on 300 *E.coli* promoters and 300 non-promoter random sequences. Authors have not announced what length of sequences was examined. GCS achieved better accuracy than individual KBANN grammar and combined grammars, and better specificity than WANN approach.

7 Summary

GCS was found useful in finding and representing *E.coli* promoter region. The grammar-based classifier system provided comparable or better results to the specialized formal system based on human-devised domain theory and knowledge discovered by neural network learning. It is worth mentioning that the proposed approach does not break up promoter regions into important or unimportant parts (such as *contact*, *conformation*, *minus_35*, *minus_10*), but treats them as whole entities. Therefore this method could be preferable in cases when we have sufficient number of known promoter regions, but might not know anything about their composition. The results suggest that the information in “unimportant” parts (gaps) might also be important for right recognition.

References

1. Angluin, D.: Queries and concept learning. *Machine Learning* 2(4) (1988) 319-342
2. Blattner, F., Plunkett, G., Bloch, C., Perna, N., Burland, V., Riley, M., Collado-Vides, J., Glasner, J., Rode, C., Mayhew, G. et al.: The complete genome sequence of *Escherichia coli* K-12. *Science*, 277 (1997) 1453-1462
3. Collado-Vides, J.: Grammatical model of the regulation of gene expression. *Proc. Natl. Acad. Sci. USA*, 89 (1992) 9405-9409
4. Dupont, P.: Regular Grammatical Inference from Positive and Negative Samples by Genetic Search. In: *Grammatical Inference and Application, Second International Colloquium ICG-94*, Berlin, Springer (1994) 236-245
5. Durbin, R., Eddy, S., Krogh, A., Mitchison, G.: *Biological Sequence Analysis*. Cambridge University Press, Cambridge (1998)
6. Gold, E.: Language identification in the limit. *Information Control* 10 (1967) 447-474
7. de la Higuera, C.: Current trends in grammatical inference. In: Ferri, F.J. et al. (eds.): *Advances in Pattern Recognition. Joint IAPR International Workshops SSPR+SPR'2000*, LNCS 1876, Springer (2000) 28-31
8. Holland, J.: *Adaptation*. In: Rosen, R., Snell, F.M. (eds.): *Progress in theoretical biology*. New York, Plenum (1976)
9. Huijzen, W.: *Genetic Grammatical Inference: Induction of Pushdown Automata and Context-Free Grammars from Examples Using Genetic Algorithms*. M.Sc.-thesis, Dept. of Computer Science, University of Twente, Enschede, The Netherlands (1993)

10. Kammeyer, T.E., Belew, R.K.: Stochastic Context-Free Grammar Induction with a Genetic Algorithm Using Local Search. Technical Report CS96-476, Cognitive Computer Science Research Group, Computer Science and Engineering Department, University of California at San Diego (1996)
11. Keller, B., Lutz, R.: Evolutionary induction of stochastic context-free grammars. *Pattern Recognition Journal* 38 (2005) 1393-1406.
12. Korkmaz, E.E., Ucoluk, G.: Genetic Programming for Grammar Induction. In: Proc. of the Genetic and Evolutionary Conference GECCO-2001, San Francisco Ca, Morgan Kaufmann Publishers (2001)
13. Lankhorst, M.M.: A Genetic Algorithm for the Induction of Nondeterministic Pushdown Automata. Computing Science Reports CS-R 9502, Department of Computing Science, University of Groningen (1995)
14. Lanzi, P.L., Riolo, R.L.: A Roadmap to the Last Decade of Learning Classifier System Research. LNAI 1813, Springer Verlag (2000) 33-62
15. Lee, L.: Learning of Context-Free Languages: A Survey of the Literature. Report TR-12-96. Harvard University, Cambridge, Massachusetts (1996)
16. Leung, S.W., Mellish, C., Robertson, D.: Basic gene grammars and dna-chart parser for language processing of Escherichia coli promoter dna sequences. *Bioinformatics*, 17 (2001) 226-236
17. Lewin, B.: *Genes VII*. Oxford University Press, Oxford (2000)
18. Losee, R.M.: Learning Syntactic Rules and Tags with Genetic Algorithms for Information Retrieval and Filtering: An Empirical Basis for Grammatical Rules. In: *Information Processing & Management* (1995)
19. Lucas, S.: Context-Free Grammar Evolution. In: *First International Conference on Evolutionary Computing* (1994) 130-135
20. Mishra, R., Chatterji, D.: Promoter search and strength of a promoter: two important means for regulation of gene expression in Escherichia coli. *J. Biosci.*, 18 (1993) 1-11
21. Murphy, P.M., Aha D.W.: UCI Repository of Machine Learning Databases. Department of Information and Computer Science, University of California at Irvine, Irvine, CA (1992)
22. O'Neill, M.: Escherichia coli promoters: neural networks develop distinct descriptions in learning to search for promoters of different spacing classes. *Nucleic Acids Res.*, 20 (1992) 3471-3477.
23. Pereira, F., Warren, D.: Definite clause grammars for language analysis. *Artif. Intell.*, 13 (1980) 231-278
24. Rice, P., Elliston, K., Gribskov, M.: DNA. In: Girbskov, M., Devereux, J. (eds.): *Sequence Analysis Primer*, chapter 1, Stockton Press (1991) 1-59
25. Rosenblueth, D., Thieffry, D., Huerta, A., Salgado, H., Collado-Vides, J.: Syntactic recognition of regulatory regions in Escherichia coli. *Comput. Appl. Biosci.*, 12 (1996) 415-422
26. Sakakibara, Y: Grammatical Inference in Bioinformatics. *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 27 no. 7 (2005) 1051-1062
27. Searls, D.: Investigating the linguistics of DNA with definite clause grammars. In: Lusk, E. R., O. (eds.): *Logic Programming: Proceedings of the North America Conference on Logic Programming*, vol. 1, Association for Logic Programming (1989) 189-208
28. Searls, D.: The linguistics of DNA. *Am. Sci.*, 80 (1992) 579-591
29. Searls, D.: The computational linguistics of biological sequences. In: Hunter, L. (ed.): *Artificial Intelligence and Molecular Biology*, chapter 2, MIT Press, Boston, MA (1993) 47-120

30. Searls, D.: Linguistic approaches to biological sequences. *Bioinformatics*, 13 (1997) 333–344
31. Shavlik, J., Towell, G., Noordewier, M.: Using neural networks to refine existing biological knowledge. *Int. J. Genome Res.*, 1 (1992) 81–107
32. Smith, T.C., Witten, I.H.: Learning Language Using Genetic Algorithms. In: Wermter, S., Rilo, E., Scheler, G. (eds.): *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*, LNAI 1040 (1996)
33. Stolzmann, W.: *An Introduction to Anticipatory Classifier Systems*. LNAI 1813. Springer-Verlag. (2000) 175-194
34. Towell, G., Shavlik, J.: Extracting refined rules from knowledge-based neural networks. *Machine Learning* 13 (1993) 71–101
35. Unold, O.: Context-free grammar induction using evolutionary methods, *WSEAS Trans. on Circuits and Systems*, 3 (2) (2003) 632-637
36. Unold, O., Cielecki, L.: Grammar-based Classifier System. In: Hryniewicz, O. at all (eds.) *Issues in Intelligent Systems: Paradigms*. EXIT Publishing House, Warsaw (2005) 273-286
37. Unold, O.: Playing a toy-grammar with GCS. In Mira J., Álvarez J.R. (eds.) *IWINAC 2005*, LNCS 3562 (2005) 300-309
38. Unold O.: Context-free grammar induction with grammar-based classifier system. *Archives of Control Science*, vol. 15 (LI) 4 (2005) 681-690
39. Wilson, S.W.: Classifier Fitness Based on Accuracy. *Evolutionary Computation* 3 (2) (1995) 147-175
40. Wyard, P.: Context Free Grammar Induction Using Genetic Algorithms. In: Belew, R.K., Booker, L.B. (eds.): *Proceedings of the Fourth International Conference on Genetic Algorithms*, San Diego, CA. Morgan Kaufmann (1991) 514—518
41. Zhou, H., Grefenstette, J.J.: Induction of Finite Automata by Genetic Algorithms. In: *Proceedings of the 1986 International Conference on Systems, Man and Cybernetics* (1986) 170-174
42. Younger, D.: Recognition and parsing of context-free languages in time n^3 . University of Hawaii Technical Report, Department of Computer Science (1967)

Learning Bayesian Classifiers from Dependency Network Classifiers

José A. Gámez, Juan L. Mateo, and José M. Puerta

Computing Systems Department
Intelligent Systems and Data Mining Group – $i^3\mathcal{A}$
University of Castilla-La Mancha
Albacete, 02071, Spain
jgamez@dsi.uclm.es

Abstract. In this paper we propose a new method for learning Bayesian network classifiers in an indirect way instead of directly from data. This new model is a classifier based on *dependency networks* [1] that is a probabilistic graphical model similar to Bayesian networks but in which directed cycles are allowed. The benefits from doing things in this way are that learning process for dependency networks can be easier and simpler than learning Bayesian networks, with the direct consequence that learning algorithms could have good properties about scalability. We show that it is possible to take advantage of this facility to get Bayesian networks classifiers without losing quality in classification.

1 Introduction

In this paper we present a new approach to learn Bayesian classifiers from a representation of data in form of *dependency network classifiers* rather than from data directly. This idea is taken from [2], where it is described a method for learning general Bayesian networks (BNs) from dependency networks. Dependency networks (DNs) were proposed in [1] as a new probabilistic graphical model similar to BNs, but with a key difference: the graph that encodes the model structure does not have to be acyclic. As in BNs each node in a DN has a conditional probability distribution given its parents in the graph. Although this feature can be observed as the capability to represent richer models, the price we have to pay is that usual BNs inference algorithms cannot be applied and Gibbs sampling has to be used in order to recover the joint probability distribution (see [1]).

Learning a DN from data is easier than learning a BN, specially because restrictions about cycles are not taken into account and parents for each variable can be discovered independently, which produces scalable algorithms. By learning BNs from DNs we can get two main advantages, as it is shown in [2]: First, learning a DN is easier and faster than learning a BN from data; and second, by transforming the obtained DN into a BN, we can use all the range of BNs inference algorithms instead of using Gibbs sampling, which in general is slower. When we focus in the task of classification the second advantage does not apply

because as all the variables are instantiated but the class, there is no difference in terms of inference between the two models. In spite of this fact, we can take advantage of ease of learning DNs.

In this work we focus on the comparison of the classifier model KDB (k -dependence Bayesian classifier [3]) with some proposal of dependency networks classifiers based on the same idea as the KDB model. For the sake of completeness we also consider other standard BN classifiers. To do this we have organized the paper as follows: In Sect. 2 DNs are briefly described. In Sect. 3 we describe the two-stage process of obtaining BN classifiers from DN classifiers learning from data. In Sect. 4 we show the experimental results for the proposed algorithms and compare them with state-of-the-art BN classifiers. In Sect. 5 we present our conclusions and some open research lines for the future.

2 Preliminaries

A Bayesian network \mathcal{B} over the domain $\mathbf{X} = \{X_1, X_2, \dots, X_n\}$, can be defined as a tuple (\mathbb{G}, P) where \mathbb{G} is a directed acyclic graph, and P is a joint probability distribution. Due to the conditional independence constraints encoded in the model, each variable is independent of its non-descendants given its parents. Thus, P can be factorized as follows:

$$P(\mathbf{X}) = \prod_{i=1}^n P(X_i | Pa_i) \quad (1)$$

A dependency network is similar to a BN, but the former can have cycles in its graph. In this model, based on its conditional independence constraints, each variable is independent of *all other variables* given its parents. A DN \mathcal{D} can be represented by (\mathbb{G}, \mathbf{P}) where \mathbb{G} is a directed graph not necessarily acyclic, and as in a BN $\mathbf{P} = \{\forall i, P(X_i | Pa_i)\}$ is the set of local probability distributions, one for each variable. In [1] it is shown that inference over DNs, recovering joint probability distribution of \mathbf{X} from the local probability distributions, is done by means of Gibbs sampling instead of the traditional inference algorithms used for BNs due to the potential existence of cycles.

A dependency network is said *Consistent* if $P(\mathbf{X})$ can be obtained from \mathbf{P} via its factorization from the graph (Equation 1). This condition is very restrictive, so in [1] the authors defined *general* DNs in which consistency is not required. General DNs are interesting for automatic learning due to the fact that each local probability distribution can be learned independently, although this local way of processing can lead to inconsistencies in the joint probability distribution. Furthermore, structural inconsistencies ($X_i \in pa(X_j)$ but not $X_j \in pa(X_i)$) can be found in general DNs. Nonetheless, in [1] the authors argued that these inconsistencies can disappear, or at least be reduced to the minimum when a reasonably large dataset is used to learn from. In this work we only consider general DNs.

With respect to inference, due to the fact that we focus on classification and under the assumption of complete data (i.e. no missing data neither in the

training set nor in the test set), Gibbs sampling can be avoided. This result comes from the fact that we can use *modified ordered Gibbs sampling* as designed in [1], where the sampling process for a variable can be avoided if the values taken by its parents are known. It is clear that this is the case in classification where MAP hypothesis is used to determine the class for a given instance.

3 Algorithms

In this Section we present our DN-based classifiers, that are based on the well known KDB classifier [3]. In KDB the idea is to extend the Naive Bayes classifier by allowing to each predictive variable to have k parents apart of the class (C). Thus, the algorithm computes the mutual information $I(X_i; C)$ for each predictive variable, and process them in decreasing order of $I(\cdot; \cdot)$. Then, when variable X_i is being processed the algorithm computes $I(X_i; X_j|C)$ for each X_j already included in the model. The k variables with highest value of $I(X_i; \cdot|C)$ are made parents of X_i .

Taking the idea behind of KDB algorithm and with the possibility of using cycles, we propose the following three DN-based Bayesian classifiers:

KDDN1.- k -dependence network classifier 1 (KDDN1) is our simplest proposal and it is a direct adaptation of KDB. Thus, for each variable X_i we pick the k variables with higher $I(X_i; \cdot|C)$ and make them parents of X_i (see Fig. 1). Notice that we do not need to sort previously the predictive variables by using $I(\cdot; C)$ and that by using KDDN1 all the predictive variables have exactly k parents (with it is not true for the first k variables considered in KDB). With this algorithm we select the more informative variables (given the class) as parents for each variable, but it is easy to see that structural inconsistencies can appear.

```
Initialize structure to Naive Bayes
```

```
For each variable  $X_i$ 
```

```
    Find  $k$  variables  $(X_{j_1}, \dots, X_{j_k})$  with the highest  $I(X_i; X_j|C)$ 
```

```
    Make these  $k$  variables parents for  $X_i$ 
```

Fig. 1. Pseudo-code for KDDN1 algorithm

KDDN2.- The second proposal, KDDN2, is more complex and tries avoid structural inconsistencies. To accomplish this, all links are added in pairs, that is, if $X_j \rightarrow X_i$ is added then $X_i \rightarrow X_j$ is also added. Of course, the addition is allowed only if $pa(X_i) < k$ and $pa(X_j) < k$, in other case, we discard X_j and look for the next variable in the ranking $I(\cdot, \cdot|C)$. Notice that in this case the order in which variables are analyzed is relevant, so we compute $I(X_i, X_j|C)$ for each pair (i, j) , s.t. $i < j$ and sort them in decreasing order (see Fig. 2).

```

Initialize structure to Naive Bayes
Compute a vector with  $I(X_i;X_j|C)$  for all pairs  $(i,j)$  s.t.  $i < j$ 
Sort vector in decreasing order

While there was variables without  $k$  parents AND elements in
vector
  Get next pair  $(X_i, X_j)$  from vector
  If  $(\text{parents}(X_i) < k)$  AND  $(\text{parents}(X_j) < k)$ 
    Add links  $X_j \rightarrow X_i$  and  $X_i \rightarrow X_j$ 

```

Fig. 2. Pseudo-code for KDDN2 algorithm

```

Initialize structure to Naive Bayes
Compute vector1 with  $I(X_i;C)$  values for all  $i=1..n$ 
Compute vector2 with  $I(X_i;X_j|C)$  values for all  $i,j=1..n; i < j$ 
Sort vector1 and vector2 decreasing

For  $i=1$  to  $n$  do
   $X_i \leftarrow$  variable in vector1[ $i$ ]
  Go to the position 1 of vector2
  While  $\text{parents}(X_i) < k$ 
    Get next pair in vector2 with  $X_i$  ( $(X_i, X_j)$  or  $(X_j, X_i)$ )
    Add new link  $X_j \rightarrow X_i$ 
    If  $(\text{parents}(X_j) < k)$  add new link  $X_i \rightarrow X_j$ 

```

Fig. 3. Pseudo-code for KDDN3 algorithm

```

 $(\mathbb{G}_D, P_D) \leftarrow$  Learn dependency network classifier from data
 $\mathbb{G}_B = \mathbb{G}_D \leftarrow$  Initialize BN structure to DN structure

 $\mathbf{E} = \{E_i \text{ s.t. link } E_i \text{ is included in a cycle}\}$ 
While  $|\mathbf{E}| \neq 0$  do
  For each link,  $E_i \in \mathbf{E}$ 
    Evaluate structure  $\mathbb{G}_B \setminus \{E_i\}$ 
  Let  $E_{\max}$  be the link with the best result
  Remove  $E_{\max}$  from  $\mathbb{G}_B$  and update  $\mathbf{E}$ 

Return  $(\mathbb{G}_B, P_B)$ 

```

Fig. 4. Pseudo-code for DN2BN algorithm

KDDN3.- The third proposal, KDDN3, is between the other two in the sense that it does not avoid all the structural inconsistencies, but tries to reduce them to the minimum. The idea is that if we decide to add link $X_i \rightarrow X_j$, then we also add $X_j \rightarrow X_i$ if possible. Again the order in which variables are considered is relevant, so we first rank the variables by using $I(\cdot, C)$ (see Fig. 3).

From DN-based Bayesian classifiers to BN classifiers.- Once we have learnt DN-based classifiers from data it is time to transform them into BN classifiers. The idea of this algorithm is taken from [2] but our approach is different. The algorithm is just a wrapper algorithm which behaves greedily by removing at each step the link (from those that belong to any cycle) whose elimination yields the best classifier. As we have mentioned our algorithm is a wrapper because the *best* classifier is measured in terms of its accuracy by using a cross validation with $k=5$ folds (see Fig. 4 for a description of the algorithm).

4 Results

To evaluate the Bayesian classifiers learned from DN classifiers and to measure their quality, we have selected a set of datasets from the UCI repository [4]. These datasets are described in Table 1. We have preprocessed the dataset in order to remove missing values (replacing by the mean or mode) and to discretize continuous variables (Fayyad and Irani [5] algorithm was used). The experimentation process consists on running each algorithm for each database in a 5x2 cross validation as described in [6]. State-of-the-art BN classifiers: KDB [3], with $k=1,2$ and 3; TAN [7] and Naive Bayes [8,9]; have been used as control algorithms in order to test our proposals. The results for these models are shown in Table 2, while Table 3 shows the results for the KDDN models and Table 4 shows the results for the Bayesian classifiers obtained from DN-based classifiers by using DN2BN algorithm.

Table 1. Description of the datasets used in the experiments

| Datasets | instances | attributes | $ C $ | continuous? | missing? |
|------------|-----------|------------|-------|-------------|----------|
| australian | 690 | 15 | 2 | yes | no |
| heart | 270 | 14 | 2 | yes | no |
| hepatitis | 155 | 20 | 2 | yes | yes |
| iris | 150 | 5 | 3 | yes | no |
| lung | 32 | 57 | 3 | no | yes |
| pima | 768 | 9 | 2 | yes | no |
| post-op | 90 | 9 | 3 | yes | yes |
| segment | 2310 | 20 | 7 | yes | no |
| soybean | 683 | 36 | 19 | no | yes |
| vehicle | 846 | 19 | 4 | yes | no |
| vote | 435 | 17 | 2 | no | yes |

Table 2. Classification accuracy for BN classifiers

| | KDB | | | NB | TAN |
|------------|-------------|------|-------------|-------------|-------------|
| | 1 | 2 | 3 | | |
| australian | 84.6 | 83.9 | 84.3 | 86.3 | 83.9 |
| heart | 81.6 | 80.7 | 79.3 | 84.2 | 81.9 |
| hepatitis | 87.9 | 86.8 | 86.5 | 85.0 | 85.3 |
| iris | 95.1 | 96.1 | 95.9 | 96.3 | 94.8 |
| lung | 48.8 | 48.1 | 41.9 | 52.5 | 52.5 |
| pima | 76.3 | 76.2 | 75.5 | 76.9 | 77.2 |
| post_op | 66.2 | 65.3 | 65.6 | 67.3 | 65.3 |
| segment | 94.2 | 92.7 | 91.6 | 90.9 | 93.7 |
| soybean | 91.6 | 87.0 | 85.7 | 90.7 | 89.9 |
| vehicle | 71.2 | 70.6 | 69.6 | 62.7 | 71.9 |
| vote | 93.6 | 94.9 | 95.0 | 90.2 | 94.5 |

Table 3. Classification accuracy for the KDDN algorithms ($k=1,2$ and 3)

| | KDDN1 | | | KDDN2 | | | KDDN3 | | |
|------------|-------|------|------|-------|------|------|-------|------|------|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| australian | 81.9 | 78.1 | 77.0 | 84.4 | 80.0 | 80.9 | 81.5 | 79.1 | 78.2 |
| heart | 80.1 | 78.0 | 74.8 | 79.9 | 77.4 | 75.0 | 81.0 | 77.6 | 73.5 |
| hepatitis | 85.4 | 84.3 | 84.1 | 85.3 | 85.3 | 84.8 | 84.4 | 83.9 | 83.4 |
| iris | 93.9 | 87.6 | 83.2 | 94.5 | 91.3 | 83.2 | 94.1 | 89.1 | 83.2 |
| lung | 51.3 | 48.8 | 40.0 | 46.9 | 49.4 | 47.5 | 42.5 | 47.5 | 41.3 |
| pima | 75.6 | 71.9 | 69.5 | 76.3 | 73.8 | 70.3 | 74.9 | 72.3 | 69.4 |
| post_op | 62.2 | 64.4 | 65.8 | 64.9 | 60.7 | 63.3 | 62.7 | 64.9 | 67.1 |
| segment | 93.4 | 92.6 | 90.5 | 93.0 | 93.9 | 91.5 | 93.0 | 92.7 | 90.9 |
| soybean | 88.8 | 84.3 | 81.4 | 88.7 | 87.3 | 86.3 | 87.6 | 84.5 | 82.4 |
| vehicle | 70.6 | 70.6 | 70.0 | 68.3 | 69.7 | 70.4 | 71.0 | 70.7 | 69.9 |
| vote | 9471 | 9338 | 9228 | 9297 | 9237 | 9186 | 9430 | 9255 | 9117 |

4.1 Analysis of the Results

Firstly we start by answering whether the models obtained by DN2BN algorithm improve the (KDDN) model used as initial point. To accomplish this task we use the Wilcoxon signed ranks test between each pair of models for each value of k . The results for these tests are shown in Table 5, and in all cases, but one, DN2BN algorithm produces significant improvements. The only case in which the improvement is not statistically significant is the one whose initial model was KDDN1 with $k = 1$, but, in spite of that, the model from DN2BN win in more databases to KDDN1-1. Therefore we can say that the DN2BN algorithm improves significantly the dependency network classifiers by transforming them into Bayesian classifiers.

The second step will be to compare the results obtained by state-of-the-art classifiers and those obtained by the models induced by DN2BN algorithm. In

Table 4. Classification accuracy for the BN classifiers learned with the DN2BN algorithm from the KDDN models ($k=1,2$ and 3)

| | DN2BN _{KDDN1} | | | DN2BN _{KDDN2} | | | DN2BN _{KDDN3} | | |
|------------|------------------------|------|------|------------------------|-------------|------|------------------------|------|------|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| australian | 84.8 | 83.0 | 83.0 | 86.1 | 86.0 | 85.5 | 84.9 | 85.1 | 84.2 |
| heart | 82.1 | 81.0 | 80.0 | 83.3 | 82.4 | 81.5 | 82.7 | 81.1 | 80.7 |
| hepatitis | 86.7 | 85.0 | 85.7 | 85.2 | 86.9 | 85.8 | 86.6 | 85.7 | 86.3 |
| iris | 94.9 | 95.9 | 96.0 | 96.1 | 95.6 | 96.0 | 94.8 | 95.6 | 96.0 |
| lung | 46.9 | 47.5 | 45.6 | 51.9 | 45.0 | 44.4 | 45.6 | 47.5 | 45.6 |
| pima | 77.6 | 76.2 | 75.4 | 77.1 | 77.7 | 77.0 | 77.6 | 75.6 | 75.2 |
| post_op | 65.3 | 65.1 | 63.8 | 68.7 | 69.3 | 66.0 | 65.3 | 64.4 | 65.3 |
| segment | 93.2 | 93.0 | 91.2 | 93.7 | 94.8 | 94.8 | 93.4 | 93.5 | 92.9 |
| soybean | 90.2 | 88.1 | 86.2 | 92.0 | 91.4 | 92.0 | 90.8 | 90.8 | 90.1 |
| vehicle | 71.4 | 70.9 | 69.8 | 66.9 | 69.4 | 71.0 | 71.4 | 70.6 | 69.9 |
| vote | 94.2 | 94.0 | 93.8 | 91.6 | 92.9 | 93.2 | 93.3 | 94.4 | 94.5 |

Table 5. Results (p-values) obtained by the Wilcoxon signed ranks test between each KDDN model and the one obtained by DN2BN algorithm. We highlight in bold those values than mean a significant difference ($\alpha = 0.05$).

| KDDN1 vs DN2BN | | | KDDN2 vs DN2BN | | | KDDN3 vs DN2BN | | |
|----------------|-------------|-------------|----------------|-------------|-------------|----------------|-------------|-------------|
| k=1 | k=2 | k=3 | k=1 | k=2 | k=3 | k=1 | k=2 | k=3 |
| 0.10 | 0.01 | 0.01 | 0.04 | 0.03 | 0.01 | 0.01 | 0.01 | 0.00 |

Table 6. Algorithms’ distribution obtained by the Nemenyi test

| | |
|---------|--|
| Group 1 | DN2BN _{KDDN2-1} , DN2BN _{KDDN2-2} , KDB-1 |
| Group 2 | DN2BN _{KDDN2-3} , DN2BN _{KDDN1-1} , DN2BN _{KDDN3-1} |
| Group 3 | DN2BN _{KDDN1-1} , DN2BN _{KDDN3-1} , TAN, Naive Bayes |
| Group 4 | KDB-2, DN2BN _{KDDN3-2} |
| Group 5 | DN2BN _{KDDN3-3} |
| Group 6 | DN2BN _{KDDN1-2} , KDB-3 |
| Group 7 | DN2BN _{KDDN1-3} |

this case we make a global comparison in order to decide whether or not all the algorithms compared exhibit the same behaviour (i.e. no significant difference), by the non-parametric Friedman test. With the results shown in Tables 2 and 4 we get that the statistic value for Friedman test is $\chi^2_F = 24.1742$, that is greater than the critic value 22.3620. Therefore, we must conclude that at least one model presents results significantly different from the others. To get more insight we carry out a post-hoc test, the Nemenyi test, which splits algorithms in groups, placing comparable (i.e. non significantly different) algorithms in the same group. The distribution obtained by Nemenyi test, ordered by accuracy, is shown in Table 6. From the statistical analysis we can conclude that two models,

DN2BN_{KDDN₂₋₁}, DN2BN_{KDDN₂₋₂}, are as good as the best Bayesian classifier in our experiments, KDB-1, although the former are slightly better.

5 Conclusions and Future Work

We have proposed a two-stage method to learn BN classifiers from data. The method takes advantage of the fact that learning DNs from data is faster than learning BNs. Furthermore, the process of learning DNs from data can be easily parallelized, which is specially useful when dealing with very large datasets. From our experiments we can conclude that state-of-the-art BN classifiers clearly outperforms the DN-based classifiers proposed in this paper. In our opinion this fact is due to the inconsistencies presented in the DN models. Because of this, we propose the second step in which BN classifiers are obtained from the DN-based models. The statistical analysis carried out over the obtained results shows that DN2BN algorithm can be at least as good as classical BN classifiers.

The main disadvantage of our approach is due to the wrapper process carried out by DN2BN algorithm, that can be inefficient in time. Therefore we plan to work on this point by constructing Bayesian classifiers with a new version of the DN2BN algorithm based on the filter approach.

Acknowledgments. This work has been partially supported by Spanish Ministerio de Ciencia y Tecnología, Junta de Comunidades de Castilla-La Mancha and European Social Fund under projects TIN2004-06204-C03-03 and PBI-05-022.

References

1. Heckerman, D., Chickering, D.M., Meek, C., Rounthwaite, R., Kadie, C.: Dependency Networks for Inference, Collaborative Filtering, and Data Visualization. *Journal of Machine Learning Research* **1** (2000) 49–75
2. Hulten, G., Chickering, D.M., Heckerman, D.: Learning Bayesian Networks From Dependency Networks: A Preliminary Study. In: *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*. (2001)
3. Sahami, M.: Learning Limited Dependence Bayesian Classifiers. In: *Second International Conference on Knowledge Discovery in Databases*. (1996) 335–338
4. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI Repository of machine learning databases (1998) <http://www.ics.uci.edu/~mlearn/MLRepository.html>.
5. Fayyad, U., Irani, K.: Multi-interval discretization of continuous-valued attributes for classification learning. In Morgan, Kaufmann, eds.: *International Joint conference on Artificial Intelligence (IJCAI)*. (1993) 1022–1029
6. Dietterich, T.G.: Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation* **10** (1998) 1895–1923
7. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian Network Classifiers. *Machine Learning* **29** (1997) 131–163
8. Duda, R.O., Hart, P.E.: *Pattern classification and scene analysis*. John Wiley and Sons (1973)
9. Langley, P., Iba, W., Thompson, K.: An Analysis of bayesian Classifiers. In: *Proceedings of the 10th National Conference on Artificial Intelligence*. (1992) 223–228

Determining the Dependency Among Clauses Based on Machine Learning Techniques

Mi-Young Kim

School of Computer Science and Engineering
Sungshin Women's University, Korea
miykim@sungshin.ac.kr

Abstract. The longer the input sentences, the worse the syntactic parsing results. Therefore, a long sentence is first divided into several clauses, and syntactic analysis for each clause is performed. Finally, all the analysis results are merged into one. In the merging process, it is difficult to determine the dependency among clauses. To handle such syntactic ambiguity among clauses, this paper proposes two-step clause-dependency determination method based on machine learning techniques. We extract various clause-specific features, and analyze the effect of each feature on the performance. For the Korean texts, we experiment using four kinds of machine-learning methods. Logitboosting method performed best and it also outperformed the previous rule-based methods.

1 Introduction

Syntactic analysis based on a dependency structure is popular in relatively free word-order languages – such as Korean and Japanese. To construct a dependency structure is to find the governor-dependent relation for each word. The longer the input sentences, the worse the syntactic analysis results, since syntactic ambiguity increases drastically. So, a long sentence is first segmented into clauses. After segmentation of a long sentence, syntactic analysis for each clause is performed. Then, finally all the analysis results are merged into one. In the merging process, we must determine the dependency relation among clauses. Many ambiguities exist in determining the dependency among clauses, and it is a difficult issue in a dependency structure. Therefore, this paper proposes two-step clause-dependency determination method. We extract clause-specific features in each clause, and employ several machine learning algorithms for determining the real governors of clauses. We also analyze which features have good effects on the performance, and show that our proposed machine learning-based method is effective in the syntactic analysis among clauses.

This paper is organized as follows. Section 2 presents previous work on determining the dependency among clauses. In section 3, the features for machine learning will be explained. In section 4, some experimental results will show that the proposed machine learning-based method is effective in determining the dependency of clauses. Finally a conclusion will be given.

2 Previous Work

Several studies have focused on the dependency problem among clauses [1], [2], [3], [4], [5], [6]. In the syntactic analysis, the dependency ambiguity among clauses is a serious problem. So, many researches have tried to solve the dependency problem by detecting the scope embedding preference of clauses. If the scope of a clause 'A' is embedded within the scope of a clause 'B', then we determine that 'B' is the governor of the clause 'A'.

Shirai[1] manually extracted 54 final function words of Japanese clauses and classified them into 3 categories according to the embedding relation of their scopes. Based on the human linguistic knowledge, he decided the scope of 3 types (type A < type B < type C). However, many exceptional cases exist because the categories are too coarse. Noma[4] classified clauses into 10 clause types according to the endings of the predicates of clauses. Noma[4] showed the table that indicates the embedding possibility between two clauses. For the English language, Roh[5] also applied heuristic rules to determine the dependency among clause. These rules were constructed according to the comma and conjunction information.

Utsuro[6] applied machine learning method for the dependency of clauses. He employed decision list algorithm using four features -- punctuation, grammatical tag, conjugation form of the final conjugative word, and lexicalized form of a clause. It showed good performance, but no comparison was shown with the performance of the previous rule-based methods. There was also no analysis about the significance of features on the performance.

Kawahara[2] also employed Decision List method for the dependency among subordinate clauses. He only used two features – surface form of the ending of the final word in a clause and comma information. While the previous Shirai[1] and Minami[3] classified the surface forms into several classes, Kawahara[2] used the original surface forms as a feature. The reason is the previous classifications were too coarse to handle a strength order among classes precisely.

Previous works usually used rule-based methods. So, they have limits that they could not cope with many exceptional cases. Some previous researchers also applied machine learning methods. However, they did not show which feature is most effective on the performance, and which algorithm among machine learning techniques works better. Therefore, various experiments using machine learning techniques are needed, and the effect of each feature on the performance must be analyzed.

To determine the dependency among clauses, we propose two-step clause-dependency determination method based on machine learning techniques. We use various features including semantic and clause-specific features. We also compare our performance with those of previous methods.

3 The Procedure to Determine the Dependency Among Clauses

The dependency among clauses is determined through two steps. In the next subsection, we explain these two steps in detail.

Table 1. Surface form types that cannot be the governor of a dependent clause

| | Surface form of the ending of a clause | Types of governor candidate clauses | | | | | | | | |
|----------------------------|--|-------------------------------------|---------------------------|-----------------------------|------------------------|-----------------------------|---------------------------|---------------------------|--------------------------|-------------------------|
| | | ~hamyeon-seo (<i>while</i>) | ~had-aga (<i>while</i>) | ~hae-seo (<i>because</i>) | ~hamyeon (<i>if</i>) | ~han-ika (<i>because</i>) | ~hae-do (<i>though</i>) | ~ha-ji-man (<i>but</i>) | ~han-unde (<i>and</i>) | ~hanun (<i>while</i>) |
| Types of dependent clauses | ~hamyeon-seo | X | | | | | | | | |
| | ~hadaga | X | X | | | | | | | |
| | ~haeseo | | | X | | | | | | |
| | ~hamyeon | X | X | X | X | | | | | |
| | ~hanika | X | X | X | X | X | X | | | X |
| | ~haedo | X | X | X | X | | X | | | |
| | ~hajiman | X | X | X | X | X | X | | X | |
| ~hanunde | X | X | X | X | X | X | X | X | | |

X : cannot be the governor.

3.1 Two Steps in Determining the Dependency Among Clauses

Step 1: Reducing the governor candidates of a clause

Since Korean is a head final language, the governor of a word is located in the right side of the word. So, when we select the governor of a clause, all the clauses in the right side of the clause are considered governor candidates. Therefore it needs to reduce the number of governor candidates. (From now, we call the clause that will be assigned one governor clause ‘dependent clause’.)

From the rules that Noma[4] introduced, we collected and modified the information of the Korean clause types that cannot embed a certain type of a ‘dependent clause’. It means we can recognize the clauses that cannot be the governor of a certain ‘dependent clause’.

Table 1 shows the types of the clauses that cannot be the governor of a ‘dependent clause’, according to the surface form of the ending of the predicate in a ‘dependent clause’. Using the information in Table 1, we first exclude the clauses that cannot be the governor of a ‘dependent clause’ from the governor candidates. However, Table 1 is only for 8 kinds of surface forms of the endings of ‘dependent clause’s. Therefore, it does not have a broad coverage.

Step 2: Machine learning techniques to determine the dependency among clauses

To determine the dependency among clauses, we apply various machine learning algorithms --such as Decision trees, Support vector machines, and two boosting methods. Decision tree induction algorithms have been successfully applied to NLP problems, such as parsing [7], discourse analysis[8], and word segmentation[9]. SVM has also been used in many NLP applications [10], [11], [12]. SVMs have good characteristics to cope with the data sparseness problem and achieve high generalization even with training data of a very high dimension. Boosting methods also show good performance for clause detection[13] and parsing[14]. The dependency decision

problem among clauses requires a multi-class classifier. Since SVMs are binary classifiers, we must extend SVMs to work as multi-class classifiers. As machine learning programs, we employed LIBSVM[15] for multi-class. For a decision tree algorithm, we employed a C4.5[16] decision tree induction program. For boosting, we used a Adaboost.M1[17] based on decision trees and Logitboost[18] using a decision stump as a base weak classifier.

3.2 Features

Since the head word of a clause is a predicate, we mainly use predicate information for the feature set of machine learning techniques. That is, feature extraction is done by focusing on predicates of clauses.

Each clause has 5 features, as shown in Table 2. The 1st feature concerns the POS-tag for a content word of a predicate. A content word can be assigned one of 12 kinds of POS-tags, as shown in Table 3. The 2nd feature takes the value of the surface form of the last ending of a predicate. Korean is an agglutinative language and the ending of a predicate indicates the connective function with the next clause. So, the surface form of the ending can have an influence on the clause dependency. As a 3rd feature, a semantic concept of a predicate is used. A semantic concept expressed by the Kadokawa thesaurus is divided into 1110 semantic classes. The 4th feature is a clause-specific feature. It indicates whether a clause shares the same subject with a ‘dependent clause’ or not. When clauses require the same subject, the subject appears only once in a sentence and clauses share the subject. In many cases, the real governor of a ‘dependent clause’ is the nearest clause. However, if the nearest clause does not share the same subject with a ‘dependent clause’, then the real governor can be a far clause that shares the same subject with the ‘dependent clause’. So, the subject-sharing information is important. The 5th feature deals with information on whether a predicate is followed by a comma or not. The use of a comma to insert a pause in a sentence is an important key in determining the embedding scope of clauses. So, it influences the dependency among clauses. The detailed values of each feature type are summarized in Table 3.

Table 2. Linguistic feature types used for learning

| | |
|-------------------------|---|
| 1 st feature | POS-tag for a content word of a predicate |
| 2 nd feature | Surface form of the last ending of a predicate |
| 3 rd feature | Semantic concept number of a predicate |
| 4 th feature | Whether a clause shares the same subject with a ‘dependent clause’ or not |
| 5 th feature | Whether comma is followed or not |

We use the information of 7 clauses– one is for a ‘dependent clause’, and the other 6 clauses are for the near clauses on the right side of the ‘dependent clause’. Here, the 6 clauses mean those that were not extracted from the governor candidates in step 1.

The class set consists of 6 values (1~6) to indicate the position of the real governor clause.

Table 3. Values for each feature type

| Feature type | Values |
|-----------------|--|
| 1 st | common noun, propernoun, dependent noun, person pronoun, reflexive pronoun, cardinal number, ordinal number, referent verb, other verb, referent adjective, other adjective, other |
| 2 nd | <i>n, ntey, ntul, ncuk, nci, lci, lcini, kena, keniwa, kenmanun, key, ko, na, nuntey, nunci, ni, nikka, taka, telato, tenci, ...</i> (all surface forms of Korean endings of predicates) |
| 3 rd | Kadokawa thesaurus concept numbers |
| 4 th | 1, 0 |
| 5 th | 1, 0 |

4 Experimental Evaluation

The experiment to determine the dependency among clauses was evaluated under the KIBS¹ data set of 23,330 Korean sentences, average 14.08 words/sentence, using 10-cross validation.

In our experiment, boosting methods performed better than those just using one classifier. Especially Logitboost showed best results among four machine learning methods. We think the reason is as follows. When we make training data manually, a lot of noise exists because training data size is big and many classes exist. Adaboost is weak in class noise. So, Logitboost is more robust than Adaboost [14]. In addition, in the multiclass problem, Logitboost is known for showing a better performance than Adaboost[5]. In the experiments using one classifier, SVM performed better than Decision Trees. As mentioned before, SVMs have good characteristics to cope with the data sparseness problem and achieve high generalization even with training data of a very high dimension.

We experimented focusing on the following five things.

1. Our two-step method vs. one-step method(using only 2nd step)
2. Decision Trees vs. SVM vs. Adaboost vs. Logitboost performance
3. Performance change when one kind of feature is removed
4. Using original 'surface form' information vs. Classifying 'surface form' into 5 classes (adnominal, final, coordinate, subordinate, quotative)
5. Previous rule-based methods vs. Our proposed method

In the experiments, we obtained the following results.

1. Two-step clause-dependency determination method by reducing the number of governor candidates achieved an improvement in the performance of about 3%.(see Table 5).
2. The maximum precision is 85.23% using Logitboosting algorithm. (see Table 5)

¹ Korean Information Base provided by KOTERM/KAIST.

3. The most significant feature is the surface form of the ending of a predicate. However, the semantic concept and POS-tag information is not effective on the performance. (see Table 6)
4. When we use the 5 class (adnominal, final, coordinate, subordinate, quotative) information instead of surface form information, the performance becomes worse. (see Table 6)
5. Machine learning-based methods outperformed the previous rule-based methods (see Table 4 and Table 5)

Table 4. Precision of rules in determining the dependency among clauses

| | Nearest principle | modifiee | Noma[4]’s rule |
|-----------|-------------------|----------|----------------|
| Precision | 68.62% | | 73.09% |

Table 5. Precision of Machine learnings in determining the dependency among clauses

| | Decision Trees | SVM | Ada-boosting | Logit-boosting |
|--|----------------|--------|--------------|----------------|
| Precision without Reducing candidates(one-step method) | 77.95% | 78.33% | 80.54% | 82.01% |
| Precision after Reducing candidates (two-step method) | 80.28% | 82.73% | 83.85% | 85.23% |

Table 6. Precision change when one kind of feature is removed

| Features | Precision change |
|---|------------------|
| Using all 5 features | 0.00% |
| 4 features without POS-tag | +1.06% |
| 4 features without surface form | -2.69 % |
| 4 features without semantic info. | +1.43% |
| 4 features without subject sharing info. | -1.95% |
| 4 features without comma | -2.16% |
| 5 features (Instead of surface form, using 5 class information) | -2.10% |

As shown in Table 6, the semantic concept and POS-tag information has bad effect on the performance. In other words, when we determine the dependency among clauses, the semantic concept and POS-tag of a content word are not important. However, the other 3 features – surface form, subject-sharing information, comma – have a good influence on determining the dependency among clauses. When using 4 features without surface form information, the experiment showed the biggest decrease of precision. Namely, the most effective feature is the surface form of the ending of a predicate. As mentioned before, the surface form information indicates the connective function with the next clause (e.g. ‘*umulo*(because)’ indicates it functions as a reason for the next clause). So, it contributes most significantly to the dependency among clauses.

To compare our performance with those of the previous methods, we performed two other experiments. One is the experiment based on the ‘nearest modifier principle’. This principle determines that the governor clause is the right near clause of a ‘dependent clause’. The other experiment is based on Noma[4]’s rule. As described in Table 4 and Table 5, Machine-learning based methods showed better performance.

As shown in the last row of Table 6, the original surface form information is more effective than the class information, because classes are too coarse and cause information loss.

5 Conclusion

This paper described a two-step method to determine the dependency among clauses based on machine learning techniques. In the first step, we reduce the number of governor candidates. In the second step, the procedure to determine the dependency among clauses is performed based on various machine learning methods. We also extracted 5 kinds of features in a clause and analyzed the effect of each feature on the performance.

The experimental results show that the proposed method is effective in determining the dependency among clauses, and Logitboosting method shows the best performance, with the precision of 85.23 %. We also analyze that surface form of the ending of a predicate, comma, and subject-sharing information contribute to the performance. However, semantic information and POS-tag features decrease the performance.

We plan to continue our research as follows. First, we will analyze the clause-dependency errors after applying the proposed method, and try to improve the clause-dependency performance. Second, we will apply this method to other languages, because most languages have difficulty in determining the dependency among clauses.

Acknowledgements. This work was supported by the Sungshin Women's University Research Grant of 2007.

References

1. S. Shirai, S. Ikehara, A. Yokoo, and J. Kimura. “A new dependency analysis method based semantically embedded sentence structures and its performance on Japanese subordinate clauses.” *Transactions of Information Processing Society of Japan*, 36(10):2353-2361, 1995
2. D. Kawahara and S. Kurohashi. *Corpus-based Dependency Analysis of Japanese Sentences using Verb Bunssetsu Transitivity*, In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium*, pp. 387-391, 1999
3. F. Minami. “*Gendai Nihongo no Kouzou*’(structures of Modern Japanese Language), Tai-shuukan shoten, 1974
4. Noma Hideki, “The relations between Korean surface forms and grammars (in Korean)”, 2002

5. Yoon-Hyung Roh, Young Ae Seo, Ki-Young Lee, Sung-Kwon Choi: Long Sentence Partitioning using Structure Analysis for Machine Translation. Proceeding on NLPRS, p.646-652, 2001.
6. T. Utsuro, S. Nishiokayama, M. Fujio, and Y. Matsumoto, "Analyzing dependencies of Japanese subordinate clauses based on statistics of scope embedding preference," Proc. 1st Conference of the North. American Chapter of the ACL, pp.110-117, 2000
7. M. Haruno, S. Shirai, and Y. Ooyama, "Using decision trees to construct a practical parser", Proc. 36th Annual Meeting of the Association for Computational Linguistics, Monteval, Quebec, Canada , pp.505-511, 1998.
8. T. Nomoto and Y. Matsumoto. "Discourse parsing: a decision tree approach", Proc. 6th Workshop on Very Large Corpora, Montreal, Quebec, Canada, pp.216-224, 1998
9. V. Sornertlamvanich, T. Potipiti and T. Charoenporn, "Automatic corpus-based Thai word extraction with the C4.5 learning algorithm", Proc. 18th International Conference on Computational Linguistics, Saarbrucken, Germany , pp.802-807, 2000.
10. T. Kudo and Y. Matsumoto, "Chunking with Support Vector Machines", Proc. 2nd meeting of North American Chapter of Association for Computational Linguistics (NAACL), Pittsburgh, PA, USA, pp.192-199, 2001
11. T. Kudo and Y. Matsumoto, "Japanese Dependency Structure Analysis Based on Support Vector Machines", Proc. 2000 SIDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000), Hongkong, China, pp.18-25, 2000
12. H. Yamada and Y. Matsumoto, "Statistical Dependency Analysis with Support Vector Machines", Proc. 8th International Workshop on Parsing Technology, Nancy, France, pp.195-206, 2003
13. X. Carreras and L. Marquez, "Boosting Trees for Clause Splitting", Proc. CoNLL-2001, Toulouse, France, pp.73-75, 2001
14. J. C. Henderson and E. Brill, "Bagging and Boosting a Treebank Parser", Proc. NAACL-2000, Seattle, Washington, USA, pp.34-41, 2000
15. Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
16. J. R. Quinlan. C4.5 Programs for Machine Learning. Morgan Kaufmann Publishers. 1993
17. Y. Freund and R. E. Schapire, "Experiments with a New Boosting Algorithm", In Machine Learning: Proc. 13th International Conference, Bari, Italy, pp.148-156, 1996
18. J. Friedman, T. Hastie and R. Tibshirani, "Additive Logistic Regression: A Statistical View of Boosting", Annals of Statistics, Vol.28(2), pp.337-374, 2000

Using Real-Valued Meta Classifiers to Integrate and Contextualize Binding Site Predictions

Mark Robinson, Offer Sharabi, Yi Sun, Rod Adams, Rene te Boekhorst,
Alistair G. Rust, and Neil Davey

University of Hertfordshire, College Lane, Hatfield,
Hertfordshire AL10 9AB, Great Britain

{o.Sharabi,y.2.sun,m.robinson,r.g.adams,R.teBoekhorst,
N.Davey}@herts.ac.uk, Arust@systemsbiology.org

Abstract. Currently the best algorithms for transcription factor binding site predictions are severely limited in accuracy. However, a non-linear combination of these algorithms could improve the quality of predictions. A support-vector machine was applied to combine the predictions of 12 key real valued algorithms. The data was divided into a training set and a test set, of which two were constructed: filtered and unfiltered. In addition, a different “window” of consecutive results was used in the input vector in order to contextualize the neighbouring results. Finally, classification results were improved with the aid of under and over sampling techniques. Our major finding is that we can reduce the False-Positive rate significantly. We also found that the bigger the window, the higher the F-score, but the more likely it is to make a false positive prediction, with the best trade-off being a window size of about 7.

1 Introduction

In this paper, we investigate the effect of contextualizing data, within the framework of improving the identification of transcription factor binding sites on sequences of DNA using a Support Vector Machine (SVM). There are several algorithms to search for binding sites in current use [1]. However, most of them are severely limited in their accuracy and yield many false positive results. That imposes a serious problem for practicing biologists, as experimentally validating a prediction is costly.

In [2] we attempted to reduce these false positive predictions using classifications techniques employed in the field of machine learning. Since the data is exceptionally skewed (about 93 percent are in one class, not part of a binding site), we further dealt with the problem of classification in an imbalanced data set in [3]. Although we contextualized the data in previous work [2], the window size was fixed at 7. In this paper we extend this analysis for different sizes of windows. The change in outcome is reflected by a variety of performance metrics.

2 Problem Domain

There is currently a considerable research focus towards gaining a functional understanding of genomic regulatory control. Many important biological systems are

controlled, to some extent, by Gene Regulatory Networks (GRN's) and an increased understanding of their regulation and encoding would be invaluable. Transcriptional control of gene regulation is a fundamental feature of GRN's, especially so in developmental GRN's. A crucial step for increasing our understanding of processes at this level is to be able to predict the short sequences of DNA that bind transcription factors (TFBS). These sequences effectively determine the set of proteins which are able to influence the expression of a particular gene. The computational prediction of TFBS is a necessary precursor for a genome wide analysis of GRN's.

The development of algorithms for the prediction of TFBS is a difficult problem. The rules that determine the specific set of sequences which a transcription factor will bind strongly with are both non-trivial and still not fully understood. In spite of considerable improvements in the accuracy of such algorithms in recent years, the high number of false positive predictions still severely limits the utility of such algorithms. Working on the premise that algorithms with differing algorithmic foundations may well be, to some degree, complementary, we have, in previous work, explored the use of machine learning methods for integrating 12 prediction algorithms [2]. In this work we explore the importance of data contextualization by using a "window" of neighbouring predictions as an input vector (see Sects. 3 and 5). In particular we explore the importance, if any, of the size of the window for improving prediction accuracy.

3 Description of the Data

The data is a sequence of 68910 nucleotides, each of which may be part of a binding site. For each nucleotide there is a prediction result from each of the 12 base algorithms, which may be either real valued or binary. Each nucleotide also has a label denoting whether it is part of a known binding site. The data therefore consists of 68910 12-ary real vectors, each with an associated label.

The data set was divided into a training set that consisted of 2/3 of the data, the remaining 1/3 was used as the test set. Amongst the data, there are repeated vectors, some with the same label (repeated items), and some with contradictory labels (inconsistent items). These items are unhelpful in the training set and were therefore removed. The filtered training set is called the consistent training set. However, in the case of the test set, both the full set of data and the subset of consistent test items are considered. The full data set was called the unfiltered data set, whereas the subset of consistent test items was called the filtered data set.

In the dataset, there are fewer than 10% binding positions amongst all the vectors, so this is an imbalanced dataset [4]. An imbalanced dataset imposes a problem for supervised classification algorithms, as they are expected to over-predict the majority class, namely the non binding site category. One of the techniques to overcome this problem is to apply the data based method: under-sampling of the majority class and over sampling of the minority class. For under sampling, a subset of data points from the majority class is randomly selected. For over sampling, the SMOTE algorithm is used [5]. The process of integrating, sampling and classifying the data, is illustrated in Fig. 1.

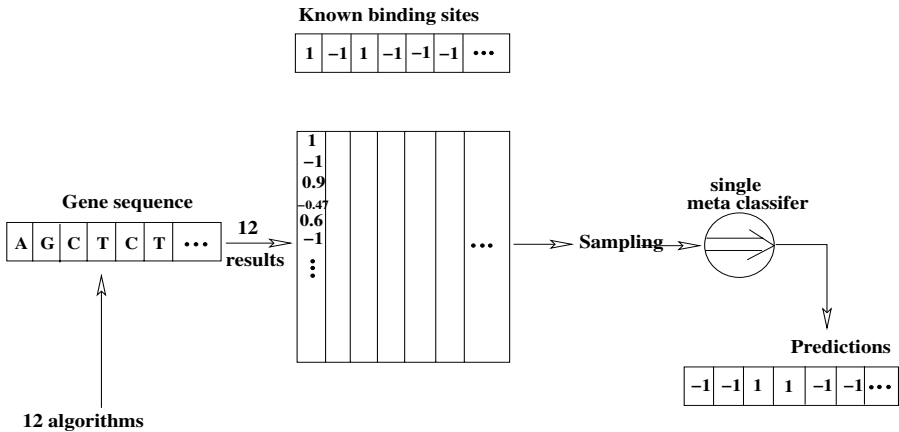


Fig. 1. The integration, sampling and classification of the data. For each location in the sequence, the prediction results of 12 the algorithms was integrated into one single vector. The data was under and over sampled, and then classified using a meta-classifier.

4 Contextualizing the Data

As the data is drawn from a sequence of DNA nucleotides the label of other near locations is relevant to the label of a particular location. In other words if a specific nucleotide is part of a binding site then it is highly likely that its neighbours will also be part of the same binding site. Therefore, adding the neighbouring vectors of a particular vector, windowing the vectors, can improve predictions. In [2] we used the location of the 3 nearest sites to either side of a given site, thereby constructing a window size of 7, and a consequent vector of 84 (12 times 7), as shown in Fig. 2.

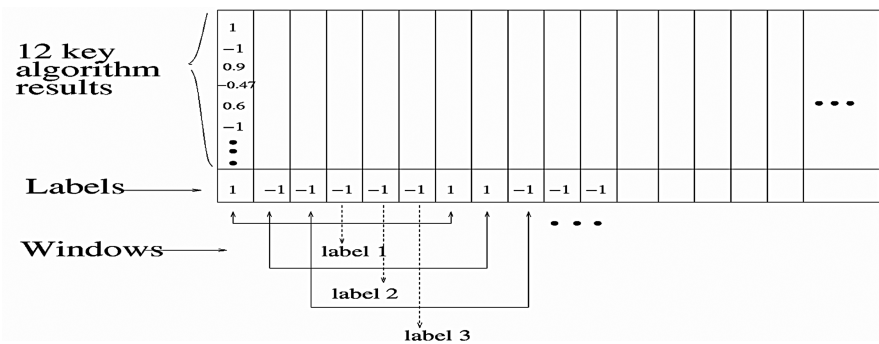


Fig. 2. Contextualising the data. In this example the window size is set to 7. The middle label prediction is the label for the windowed input. The length of each windowed input is now $12 \times (2K+1)$.

In this work, we used the location of K nearest locations to either side, where $K = (1, 2, 3, 4, 5, 6)$. The result is a window size of $2K+1$, and a consequent vector of 12 times $(2K+1)$. Windowing the vectors has the additional benefit of eliminating most of the repeated and inconsistent data. In bigger windows, not only is the training set and the test set sizes increased, but also the difference between both test sets decreased, as can be seen in Table 1.

Table 1. The sizes of *training-set*, *testing set* and *filtered testing set* used in this experiment. Note that as window size increases, the difference between the unfiltered and filtered test sets decreases.

| Window Size | Set | | |
|-------------|----------|--------------------|------------------|
| | Training | Unfiltered testing | Filtered testing |
| 3 | 17701 | 22511 | 9767 |
| 5 | 26770 | 22509 | 14399 |
| 7 | 32595 | 22507 | 17233 |
| 9 | 36670 | 22505 | 19093 |
| 11 | 39503 | 22503 | 20277 |
| 13 | 41400 | 22501 | 21064 |

5 Classifier Performance

Classification accuracy rate is not sufficient as a standard measure for a problem domain with an imbalanced data. Therefore, several common performance metrics were applied, such as *Recall* (1), *Precision* (2), *FP-rate* (3), and an *F-score* (4) [6], [7]. Also, a Receiver Operating Characteristics (*ROC*) analysis [8] was applied.

5.1 Performance Metric

Based on the confusion matrix computed from the test results, several common performance metrics can be defined, where TN is the number of True Negative samples; FP is the False Positive samples; FN is False Negative samples; TP is True Positive samples.

$$Recall = \frac{TP}{TP + FN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$FP-rate = \frac{FP}{FP + TN} \quad (3)$$

$$F\text{-score} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \quad (4)$$

Note that for all the measures except *FP-Rate* a high value is desirable. Most of the base algorithms have a high *Recall* by simply over predicting the binding site class (predicting every item to be positive gives a recall of 1), and this is problematic. On the other hand *Precision* is the proportion of the positively categorized samples that are actually part of a binding site. Increasing the *Precision* of the prediction is one of the main goals of our meta-classifier. However increasing *Precision* is normally accompanied by a decrease in the *Recall*, so the *F-score*, which takes into account both *Recall* and *Precision*, is a useful measure of overall performance. The *FP-rate* is the proportion of all the negative samples that are incorrectly predicted. The base algorithms generally have a high *FP-rate* and reducing this is another major goal of our classifier.

5.2 ROC Curves

In a ROC diagram, the true positive rate (*Recall*) is plotted on the Y-axis and the false positive rate (*FP-rate*), is plotted on the X- axis. Often, to measure a classifier performance it is convenient to use the area under an *ROC* curve (*AUC*). The *AUC* value ranges between 0 to 1, where an effective classifier should have an *AUC* which is greater than 0.5.

6 Experiments

The classification technique used in this work is a Support Vector Machine [9], and the experiments were completed using LIBSVM¹. The RBF kernel was used. The SVM therefore has two parameters, *C* (the penalty parameter) and γ (width of the kernel). The *C* values were (5, 20, 50, 100, 300, 1000, 2000) and the γ values were (0.01 0.04 0.01 0.001). The window sizes ranged from 1 to 13, in increments of 2. For each window size, the performance matrix and ROC curves of the 6 *C* values and of the 4 γ values were computed, both for the filtered test set and for the unfiltered test set. For example, for window size 5, there were altogether 48 results: 24 (6 times 4) results for the filtered set, and 24 results for the unfiltered set.

7 Results

Tables 2a and 2b contain the best results for each window size. The best results were gained when the γ value was fixed to 0.001 for all windows. The *C* value differed between the window sizes, but ranged from 5 to 300. In fact, in [10] we showed that all 4 best results for each window size were in that range, with the exception of window size 3. The results clearly show that the *FP-rate* decreased dramatically from the best algorithm to the single vector (window size 1), and decreased further when “

¹ <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.

windowing” the data. The lowest *FP-rate* was 0.005 for window size 3. Vectoring the data is useful for increased *Precision* as well. The *Precision* of window size 1 is 0.377, and higher than that of the best algorithm (0.222). The *Precision* of window-size 3 is the highest (0.649). From window size 3 onward however, the precision value slightly decreased as the window size increased. Thus, window size 3 has the lowest false positive rate, with the highest *Precision*. The AUC values of all results were higher than 0.5, meaning that the classifier performed better than chance level. Another way to determine a good result is by comparing a high F-score to a low FP-rate. In accordance with previous experiments, [1], [2], [3], there is a trade off between *F-score* and an *FP-rate*. When the *F-score* rises, so does the false positive rate. Bigger window sizes generate higher *F-scores* but also higher *FP-rates*. However, from Fig. 3, window size 7 seems to have a good trade-off between the two measures.

Table 2a. Common performance metrics calculated from confusion matrices on the *unfiltered test set*. The selected best parameters for each window size are also shown in the table.

| Window Size | <i>C</i> | γ | Recall | Precision | F-score | FP-rate | AUC |
|-------------|-----------------|----------|--------|-----------|---------|---------|-------|
| | Unfiltered data | | | | | | |
| Best Alg. | - | - | 0.400 | 0.222 | 0.285 | 0.106 | - |
| 1 | 300 | 0.001 | 0.246 | 0.377 | 0.297 | 0.031 | 0.710 |
| 3 | 5 | 0.001 | 0.111 | 0.649 | 0.190 | 0.005 | 0.650 |
| 5 | 100 | 0.001 | 0.179 | 0.504 | 0.260 | 0.013 | 0.640 |
| 7 | 50 | 0.001 | 0.197 | 0.489 | 0.280 | 0.015 | 0.650 |
| 9 | 50 | 0.001 | 0.226 | 0.446 | 0.300 | 0.021 | 0.650 |
| 11 | 20 | 0.001 | 0.231 | 0.443 | 0.300 | 0.022 | 0.660 |
| 13 | 5 | 0.001 | 0.232 | 0.430 | 0.300 | 0.023 | 0.680 |

Table 2b. Common performance metrics calculated from confusion matrices on the *filtered test set*. The selected best parameters for each window size are also shown in the table.

| Window Size | <i>C</i> | γ | Recall | Precision | F-score | FP-rate | AUC |
|-------------|---------------|----------|--------|-----------|---------|---------|-------|
| | Filtered data | | | | | | |
| 1 | 300 | 0.001 | 0.341 | 0.344 | 0.342 | 0.073 | 0.723 |
| 3 | 5 | 0.001 | 0.132 | 0.628 | 0.218 | 0.008 | 0.695 |
| 5 | 100 | 0.001 | 0.207 | 0.511 | 0.295 | 0.017 | 0.675 |
| 7 | 50 | 0.001 | 0.221 | 0.499 | 0.307 | 0.019 | 0.672 |
| 9 | 50 | 0.001 | 0.245 | 0.449 | 0.317 | 0.024 | 0.661 |
| 11 | 20 | 0.001 | 0.245 | 0.444 | 0.316 | 0.024 | 0.668 |
| 13 | 5 | 0.001 | 0.244 | 0.432 | 0.312 | 0.025 | 0.689 |

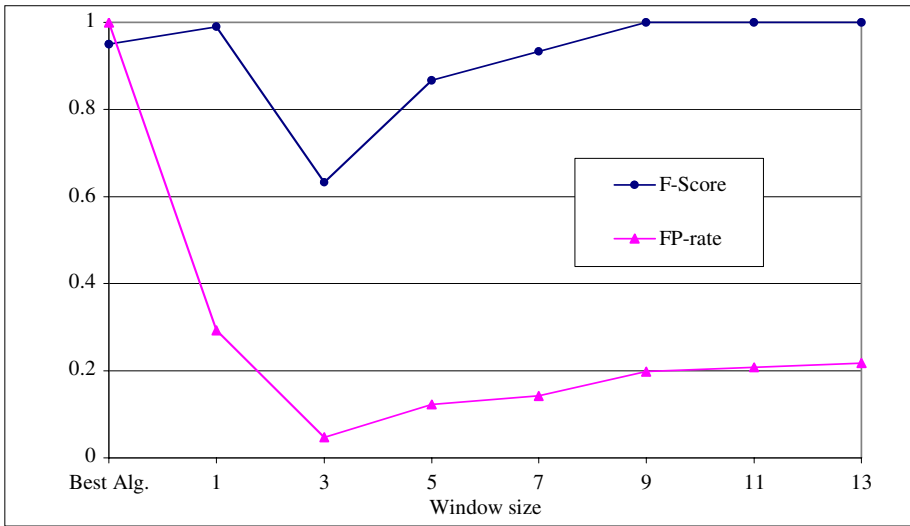


Fig. 3. Normalized (to have maximum of 1) values for *F-score* (dots) and *FP-rate* (triangles) for the various unfiltered window sizes. From window size 3 both *FP-rate* and *F-score* are increasing, as the window size increases.

8 Conclusions

An important finding of this study is that by vectoring and later “windowing” the data, the *FP-rate* is significantly decreased, from as much as 10% to as little as 0.5%. That has important implications for experimental biologists for whom the high *FP-rates* considerably reduce the utility of these algorithms. All window sizes have a better *Precision* than that of the best base algorithm, and more importantly, a better trade off between an *F-score* and an *FP-rate*. Window sizes affect the performance of the SVM classifier. The bigger the window size, the higher the *F-score*. However, that comes with a cost; the *FP-rate* is increased accordingly. Arguably, the best trade-off was gained for window size 7. The best values for the SVM parameters were fixed with the γ value on 0.001, but ranged for the *C* value from 5 to 300. For window sizes 3 to 7 the unfiltered data set had a better trade-off between *F-score* and *FP-rate*. However, as the window size got bigger, the difference between the test sets’ sizes decreased. Consequently, the similarity between the various performance metrics increased.

References

1. Robinson, M., Sun, Y., te Boekhorst, R., Kaye, P., Adams, R. & Davey, N.: “Improving computational predictions of cis-regulatory binding sites,” *Biocomputing - Proceedings of the Pacific Symposium*, (2006)
2. Sun, Y., Robinson, M., Adams, R., Kaye, P., Rust, A. G. & Davey, N.: “Using real-valued meta classifiers to integrate binding site predictions,” *Proceedings of International Joint Conference on Neural Networks*, (2005)

3. Sun, Y., Robinson, M., Adams, R., te Boekhorst, R, Rust, A. G. & Davey, N.: "Using Sampling methods to improve binding site predictions." accepted by *The 14th European Symposium on Artificial Neural Network (ESANN)*, (2006)
4. Japkowicz, N: Class imbalances: Are we focusing on the right issue? Workshop on learning from imbalanced datasets, II, ICML, Washington DC. (2003)
5. Chawla, N. V., Bowyer, K. W., Kegelmeyer L. O. & Kegelmeyer, W. P.: "SMOTE: Synthetic minority over-sampling Technique," *Journal of Artificial Intelligence Research*. Vol. 16, (2002) 321-357
6. Buckland, M. & Gey, F.: The relationship between Recall and Precision. *Journal of the American Society for Information Science*: Vol. 45, No. 1, (1994) 12--19
7. Joshi, M., Kumar, V. & Agarwal, R.: Evaluating Boosting algorithms to classify rare classes: Comparison and improvements. First IEEE International Conference on Data Mining, San Jose, CA (2001)
8. Fawcett, R.: "ROC graphs: notes and practical considerations for researchers," Kluwer Academic publishers, (2004)
9. Scholkopf, B & Smola, A. J.: *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, The MIT Press (2002)
10. Sharabi O.: "Using real-valued meta-classifiers to integrate and contextualize binding site predictions," University of Hertfordshire, STRC, technical report, in press.

Effectiveness of Feature Space Selection on Credit Engineering on Multi-group Classification Cases

Junghee Park¹, Kidong Lee², and Jinhwa Kim³

¹ Department of Business Administration,
Sogang University, Seoul, South Korea
jpark@sogang.ac.kr

² Department of Business Administration,
University of Incheon, Incheon, South Korea
kdlee@incheon.ac.kr

³ Department of Business Administration,
Sogang University, Seoul, South Korea
Jinhwakim@sogang.ac.kr

Abstract. This study tests the sensitivity of input feature space selection on credit rating using four classifiers as backpropagation(BP), Kohonen self-organizing feature map, discriminant analysis(DA), and logistic regression. The results of the study are that at individual methods applied, BP network outperforms two statistical counterparts while Kohonen network shows the least accuracy among the models. The results also show that the selection of the feature spaces to the accuracy outcome may not be very sensitive when we test the four methodologies altogether at aggregate level.

1 Introduction

Credit rating activities and resulting information seem to be conveyed critical value for the financial market participants: for security issuers, a rating affects the level of interest they have to pay and, for investors, this assigned rating tends to act as a threshold value for decisions about whether to buy the security. In sum, credit rating system serves as an investment “bridge” between different risks and earning levels for the security issuing company as well as the market investor [3], [4].

In other words, credit rating information provides a succinct index for information user to gauge the financial status or creditworthiness of a debt security. But, providing such real-time as well as accurate credit judgment to the public is a daunting job because these rating decisions themselves pretty much involve in qualitative [3], [6], [13]. Moreover, credit rating decisions are, by nature, multi-group classification problems, not like dichotomous one such as bankruptcy, making this credit rating task even more difficult.

According to Dutta and Shehker[2], evaluation activities of credit rating on a security (i.e., rating a firm’s financial strength) involve finding the following mapping function f such that

$$f: F_1 \times F_2 \times F_3 \times \dots \times F_k \rightarrow R_i. \quad (1)$$

where F be the k dimensional feature space, F_1, F_2, \dots, F_k , and R is set of (exclusive) m credit ratings, R_1, R_2, \dots, R_m . The credit rating process can be summarized in two successive steps: first, extracting the appropriate feature space ($F_1 \times F_2 \times F_3 \times \dots \times F_k$) and then finding a mapping function f in the Cartesian space ($F_1 \times F_2 \times F_3 \times \dots \times F_k$) to a rating class (R_i) [2], [8]. In prior research, the feature extraction problem has been handled by determining a relevant set of financial variables, and the second problem of finding a mapping function has been often done by utilizing contemporary statistical classifiers such as discriminant analysis [1], [3].

In this paper, we test these two successive processes of credit rating activities using US security market data described by two most frequently used feature spaces by prior researchers, Belkaoui [1], and Horrigan [5] financial variable sets. Then, we try to find a suitable mapping function to a rating class by contrasting the results of accuracy measurement of four different classification tools: two from neural network classifiers as backpropagation(BP) network, Kohonen self-organizing network, and the rest two from statistical counterpart as discriminant analysis(DA), and logistic regression served as benchmarks for neural network classifiers. Also, it is interesting to see how unsupervised neural network such as Kohonen behaves in multi-group classification contest. By comparing two widely used financial variables to four different mapping functions, it is hoped that we provide better decision making information to the field of credit rating community.

2 Prior Literature

Table 1 shows a selective research on credit rating. The credit rating question poses a typical classification or clustering problem: transformation of domain information and data (inputs) into rating symbols (outputs) using the interpretation of the rating system (processes). Also, the systems outputs (rating symbols) in turn becomes or influences on inputs to the financial system so that there is an iterative mapping process among its components and the surrounding environment [3], [5], [10], [13].

Neural network methodologies are first used in credit rating area by Dutta and Shehker[2] followed by many neural net researchers [3], [7], [8], [10], [13], mainly due to their better performance over their statistical counterparts. In fact, one problem of prior research on credit rating is however that their input feature spaces describing the underlying securities are often different one other as shown in table 1. For this reason, comparing the accuracy or other measurement results of the previous credit rating researches makes it very difficult, oftentimes useless[10], [13]. Moreover the selection of their mapping functions frequently is in inconsistency across researchers [1], [2], [8], which makes direct comparison of the measurement results across the prior studies fairly difficult. Therefore, we need some sort of common ground framework to compare credit rating research work done. In this study, we try to make a first step toward the consensus building process by comparing the two well-known and widely used financial input variable sets, Belkaoui's [1] and Horrigan's [5]. These two financial variable sets have been frequently used in the past credit rating literature [1], [2], [5], [7], [12], [13]. These two feature variable sets take somewhat different orientation toward their accountability. 6 out of 7 Belkaoui's variables[1] come from a balance sheet while three of the five Horrigan variables are from the

Table 1. Selective statistical and neural network applications applied in credit rating

| Name | Methodology | Independent (measurement) Variables | |
|-----------------|-----------------------|--|-----------------------------|
| Horriagan | Logistic | Total assets | Sales over net worth |
| | | Working capital over sales | Profit/sales |
| | | Net worth over total debt | Subordination |
| Belkaoui | DA | Total assets | Current ratio |
| | | Total debt | Fixed charge coverage ratio |
| | | Long-term debt | Short-term debt |
| | | Stock price as a book value | Subordination |
| Dutta & Shehker | BP | Liability/ (Cash+Assets) | Profit/Sales |
| | | Debt proportion | Financial strength |
| | | Sales/net worth | Earning/Fixed Costs |
| Nour | Kohonen | Debt to total assets | Interest expense to sales |
| | | (Pretax income + Interest) to Interest expense | Pretax income to net sales |
| | | Cashflow to current liability | Income to gross assets |
| | | Cash flow to total liabilities | Sales to gross assets |
| | | Total liabilities to invested capital | Sales to invested capital |
| | | | Income to market value |
| | Sales to market value | | |

income statement. So it might say that Belkaoui's research more focuses on stock approach while Horriagan's emphasizes flow side of the financial status of a debt security.

3 Research Design and Methodology

For research design and methodologies for this credit rating study, we follow two successive processes as: selecting feature space first, describing the attributes of issuing debt securities, and then, applying four different classification methodological tools for classification of the debt securities to a certain class. Dataset, experiment design, their description of input feature space, and applied methodologies are detailed below.

3.1 Data Set and Experiment Design

The data set consists of the US companies' bonds in 1994 - 1998, extracted from the S&P's Compustat North America database. The top four classes (AAA, AA, A, and BBB) are only considered since these are called the "investment-grade" that are of interest to the investor. First, 200 bonds in the 1994 - 1997 are selected: 160 of these are used for training and the remaining 40 are used for validation. Then, additional 40 selected from 1998 are used for prediction accuracy. All of three sets, training, testing and validation, contain the equal number of samples for each class. This experimental design construction does not resemble the natural distribution of outstanding ratings in the real world. Though, for experimental purposes, many rating studies adopt this equal-numbered sample distribution [7], [8], [13].

3.2 Feature Space Selection and Methodologies

As mentioned earlier, each company's bond is described by two input variable sets by Belkaoui [1] and Horrigan [5]. Belkaoui's seven variables are: total assets, total debt, long term debt/total investment capital, short term debt/total investment capital, current assets/current liabilities, net income plus total interest expense, and stock price/common equity per share. Horrigan used the following five variables as total assets, working capital/sales, net worth/total debt, sales/net worth, and net operating profit/sales. Note that Belkaoui used 7 ratios while Horrigan applied only 5. It is probably that incorporating more variable numbers in the feature space provides a better result. Thus, our initial guess may be that Belkaoui's experiment would show a better accuracy. Since we consider the top four rating classes, thus, subordination variable (0-1) indicated a legal status of the claim of a bond may not need to be included in this study. Data normalization technique will also not be considered because of the stated reasons in the bankruptcy prediction section [10], [11].

Using this experiment design, we can compare the effectiveness of one variable set over the other set directly. As stated earlier, Belkaoui's variables are static measurements, 6 out of 7 extracted from a balance sheet while Horrigan's variables, 3 of 5 ratios from an income statement, emphasize the flow concept of operating of a company during a certain period. Therefore, the comparison of these two variable sets together may give an insight which variable set (balance sheet-oriented versus income statement-oriented) is better in explaining an assigned rating and thus more accurate for classification/prediction results. In so doing, we try to compare accuracy dimension of neural net classifiers (BP and Kohonen networks) with more traditional benchmarks, statistical counterparts (DA and logistic regression), using Belkaoui's [1] versus Horrigan's variables [5].

4 Experimental Results

Table 2 gives a summary of accuracy performance of the four classification techniques applied here with the number of correct classification (training result) and the number of correct prediction in parenthesis (test result).

Table 2. Summary of performance of the four classification techniques

| Feature Space | | BP | Kohonen | DA | Logistic |
|---------------|----------|---------------|---------------|---------------|---------------|
| Belkaoui | Training | 92 (57.8%) | 70 (43.8%) | 70 (43.8%) | 89 (55.6%) |
| | Test | 19 (47.5%) | 13 (32.5%) | 17 (42.5%) | 18 (45.0%) |
| Horrigan | Training | 78 (48.8%) | 65 (40.6%) | 65 (40.6%) | 74 (46.3%) |
| | Test | 18 (45.0%) | 16 (40.0%) | 17 (42.5%) | 17 (42.5%) |

First, for the comparison of the performance of individual classifiers, BP network outperforms all three other classification techniques, whether Belkaoui’s or Horri-gan’s being used. The Kohonen self-organizing feature map shows the worst results, even below to the two other statistical benchmarks used here. Especially, the Koho-nen self-organizing network applied for the test set using Belkaoui’s shows the worst prediction accuracy at 32.5%.

It might seem that when a classification problem involves more than two clusters, the accuracy of the unsupervised learning type such as the Kohonen self-organizing network appears to drop significantly. It is probably because in the unsupervised training, the researcher does not incorporate an extra target vector, “the crucial infor-mation,” thus not only using less information than its supervised counterpart, but also not inserting *the target vector*. The two statistical classification tools showed a simil-ar performance picture but the logistic regression gives a bit better result than DA.

Table 3. χ^2 Statistics from 2X2 contingency tables

| Input Vari- able | Techniques Applied | Training set | | Test set | |
|---------------------|-----------------------|--------------|----------|----------|----------|
| | | BP | Kohonen | BP | Kohonen |
| Belkaoui | Kohonen | **3.388 | - | *4.083 | - |
| | DA | **3.388 | 0.000 | 0.325 | ***1.728 |
| | Logistic | 0.124 | ***1.846 | 0.325 | ***1.728 |
| Horri-gan | Kohonen | 1.058 | - | 0.335 | - |
| | DA | 1.058 | 0.023 | 0.335 | 0.028 |
| | Logistic | 0.002 | 1.428 | 0.876 | 0.002 |

* Significant at the 5%; ** Significant at the 10%; & ***Significant at the 20% levels;

In Table 3, we compare the performance results among these classification tech-niques using a non-parametric statistic (χ^2). When Belkaoui’ set is used, the perfor-mance result of the BP network outperforms the Kohonen network and DA in training at a 10% significance level. The BP network is also better than the Kohonen net-works in terms of the prediction accuracy at a 5% significance level. In turn, the Kohonen networks are less effective in the training set and the test set than the two other statistical classifiers at a 20% significance level. However, when Horri-gan set is used, there is no significant difference in performance either for the training and test results among the four classification choices.

In sum, the performance of individual classifiers using Belkaoui gives some dis-criminating power, for example, BP vs. Kohonen, BP vs. DA, and DA vs. Kohonen. Also, it is observed that unsupervised Kohonen network does not work very well on multi group classification problem such as credit rating. Also, statistical classifier, logistic regression gives a very comparable performance result to BP network.

When applying Horri-gan, there is no discriminating power at all across the indi-vidual methodologies applied in this study. Thus, our initial assumption of Belkaoui over Horri-gan feature sets is in verified.

Table 4. Test for the effect of the Belkaoui's versus Horrigan's input sets

| F-value or P-value | Training | Test |
|--------------------|----------|-------|
| F-value | 1.55 | .03 |
| P-value | .2599 | .8619 |

Table 4 shows the ANOVA test for the effect for the input variable set, Belkaoui's versus Horrigan's across the four methodologies altogether we applied. In Table 4, it fails to show statistical significance in accuracy performance, between Belkaoui's and Horrigan's. The F-value for the test set indicates that these two input variable sets give almost identical test results. Thus, we cannot differentiate the performance difference of Belkaoui's and Horrigan's results. If we accept this statistical result, we can say that Horrigan's approach is a bit more cost effective, since it incorporated fewer variables.

In sum, for the accuracy performance of individual classifier, BP network, traditional classifiers, and Kohonen network in that order. Especially, BP with Belkaoui variable set gives the best accuracy result. For the accuracy results between Belkaoui and Horrigan feature space, we cannot find the discriminatory power between these two variable set. It seems that Horrigan's variable set may provide more cost effective, leading to a parsimony model. It may lead to the conclusion that Horrigan's income statement approach could be more effective.

5 Summary and Conclusions

The accuracy performance of the four classification tools, BP, Kohonen, DA, logistic, is tested with two widely used feature space sets, Belkaoui's vs. Horrigan's. The sensitivity test of feature space selection in this study fails to identify the discriminatory power of the two variable sets at aggregate level when we test the four methodologies altogether. In individual tool level, however, BP network shows the most accurate results while Kohonen network give the worst results. Still, applying the unsupervised Kohonen algorithm in a multi-group classification task, such as credit rating, looks promising even though this particular study does not provide any positive input. One way to test unsupervised classifiers such as Kohonen in credit rating area is that researchers keep the track of financial status of a corporation, in various feature spaces, with other firms over time. In this way, researchers can group these firms in terms of their similarities or differences.

References

1. Belkaoui, A.: *Industrial Bonds and the Rating Process*. Greenwood Press. Westport. Connecticut. (1983)
2. Dutta, S. and Shekhar, S.: Bond Rating: A non-Conservative Application of Neural Networks. *IEEE International Conference on Neural Networks*. **5** (1988) 443-450
3. Gillespie, E.S., and Wilson, R.N.: Application of Sensitivity Analysis to Neural Network Determination of Financial Variable Relationships. *Applied Stochastic Models and Data Analysis*. **13** (1998) 409-414

4. Hand, J.R., Holthausen, R.W., and Leftwich, R.W.: The Effect of Bond Rating Agency Announcements on Bond and Stock Prices. *Journal of Finance*. **47(2)** (1992) 733-752
5. Horrigan, J.: The Determination of Long Term Credit Sharing with Financial Ratios. *Journal of Accounting Research*. **4** (1966) 44-62
6. Kim, B.O. and Lee, S.M.: A Bond Rating Expert System for Industrial Companies. *Expert Systems With Applications*. **9(1)** (1995) 63-70
7. Kim, JunWoo, Weistroffer, H. R., and Redmond, R.T.: Expert Systems for Bond Rating: A Comparative Analysis of Statistical: Rule-based and Neural Network. *Expert Systems*. **10** August (1993) 167-172
8. Nour, M.A.: Improved Clustering and Classification Algorithms for the Kohonen Self-Organizing Neural Network. Unpublished Ph.D. Dissert. Kent State University. (1994)
9. West, R.: An Alternative Approach to Predicting Corporate Bond Ratings. *Journal of Accounting Research*. **7** Spring (1970) 118-127
10. Ata Yesilyaprak.: Bond Ratings with Artificial Neural Networks and Econometric Models. *American Business Review*. **2** Jan (2004) 113
11. Jandhyala Sharma, Ravi Kamath, Sorin Tuluca.: Determinants of Corporate Borrowing: An Application of a Neural Network Approach. *American Business Review*. **21** (2003) 63
12. Katuscia Manzoni.: Modeling Eurobond Credit Ratings and Forecasting Downgrade Probability. *International Review of Financial Analysis*. **13** (2004) 277
13. Lee, Kidong, Ph.D.: Pattern Classification and Clustering Algorithms with Supervised and Unsupervised Neural Networks in Financial Applications. Kent State University. (2001)

Constructing Stereotypes for an Adaptive e-Shop Using AIN-Based Clustering

M. Virvou, A. Savvopoulos, G.A. Tsihrintzis, and D.N. Sotiropoulos

Department of Informatics
University of Piraeus
Piraeus 18534, Greece

{mvirvou, asavvop, geoatsi, dsotirop}@unipi.gr

Abstract. This paper describes an adaptive electronic video store application that monitors customers' actions and provides dynamic movie recommendation. The adaptive recommendation is formed based on double stereotypes that have been constructed for user modeling. The construction of stereotypes has been based on a novel approach that uses an Immune Network System (INS). In particular, the INS has been applied on data collected from 150 users of an earlier version of the e-commerce application. Specifically, the INS clustered users' interests as well as movies and represented each resulting cluster with corresponding antibodies. The double classification (users' interests – movies) was performed in a hierarchical way that resulted in several levels of user stereotypes: These stereotypes are then used dynamically by the e-commerce application to infer users' interests in movies based on a small set of observed users' actions.

1 Introduction

E-commerce applications have become very popular since they provide easy access to all kinds of products. However, most of existing applications are generic and do not address specific needs, preferences and attributes of individual customers. A remedy to this problem can be achieved by web personalization techniques. These techniques usually involve the construction of user models that are either based on explicit user information or on data about the user behaviour that is collected implicitly by the system. One powerful way for creating user models is based on stereotypes. Stereotypes were originally invented for the system called Grundy [8] that recommended books to users based on their preferences. In Grundy, a stereotype was defined to represent a collection of attributes that often co-occur in people and thus they enable the system to make a large number of plausible inferences on the basis of a substantially smaller number of observations.

The actual construction of stereotypes involves defining the triggering conditions (the conditions that enable a specific stereotype) and the inferences (what can be assumed for users belonging to the triggered stereotype). As Kay [5] points out there are two ways for constructing stereotypes, one is hand-crafted and the other one is empirically-based. In the hand-crafted case, the designer of the system makes assumptions about the stereotype groups whereas in the empirically-based stereotypes there is an important role of machine learning techniques in acquiring them. However,

in the majority of the existing commercial personalization systems, the personalization process involves substantial manual work and most of the time significant effort on the part of the user [7]. Apparently automating the construction of user stereotypes still remains an open research problem.

In view of the above, in our research we have introduced a novel approach of constructing user stereotypes based on an artificial immune network [4],[10] which is a clustering technique. This approach has been employed in an e-commerce application. In particular, we have developed a stereotype-based personalized e-commerce video store that is called Vision.Com and recommends movies to customers. Vision.Com aims at modeling users by silently observing their behavior rather than asking many questions to them. In this way the system aims at alleviating the users from the burden of specifying their user model by themselves.

In particular, Vision.Com was developed in two phases. In the first phase, an adaptive version of the system was created which did not incorporate any stereotypes. This version of Vision.Com was used by 150 users and all their actions within their interaction with the system were collected and analyzed by the artificial immune network. As a result of this process, in the second phase, we constructed double stereotypes that were then incorporated into the system for improving the initialization and accuracy of the user modeling component of Vision.Com. The fact that we constructed Vision.Com in two phases has provided the advantage of the construction from scratch of user stereotypes based on the clustering algorithm. In contrast, other systems (Schwarzopf [9], Ardissono and Torasso [1]) have used clustering algorithms for the revision of initial user models that had been created based on hand-crafted stereotypes.

The most relevant work to Vision.Com concerns the system developed by Cayzer and Aickelin [3] and Morrison and Aickelin [6]. That system utilized an Artificial Immune System (A.I.S) in order to tackle the task of film and web site recommendation by collaborative filtering. In that system there was a movie database called EachMovie that was used as a source of votes concerning movies that had been seen. These votes were explicitly supplied by each user and were used in order to build a recommender system that provided estimation votes for unseen movies. In contrast, in our work we obtained the movie preference related data in an implicit manner. In our approach we aim at recommending movies to a user based on his/her inferred preferences. Individual user preferences are inferred based on users' interaction with the electronic video store. In the Cayzer and Aickelin system there is neither a user model nor stereotype-based reasoning. Thus, their system cannot recommend movies that have not been seen by any user. In our case the fact that we construct user models allows the system to predict user preferences in all kinds of movies whether these have been seen by other users or not. Our aim was to develop a regulated network of antibodies that corresponded to multidimensional representative vectors of the user preferences. User preferences refer to movie attributes such as who the director of the film is, who the leading actor is, what kind of film it is etc.

In our case, the advantage of using an AIS as compared to other clustering techniques is that the construction of an Artificial Immune Network (AIN) incorporates a mutation process that applies to the customer profile feature vectors. More specifically, our work has focused on how a significant amount of redundancy within the customer profile dataset can be revealed and reduced, how many clusters intrinsic to the customer dataset can be identified and what the spatial structure of the data within the identified clusters is.

2 Adaptive e-Shop Application

Vision.Com is an adaptive e-commerce video store that learns from customers' preferences. A screenshot of Vision.Com is illustrated in Fig. 1. Its aim is to provide help to customers choosing the best movie for them. Vision.Com operates on top of the .net framework technology and uses client server techniques. The web based system runs at a local network with IIS (internet information services) playing the role of the web server.

For every user the system creates a different record at the database. In Vision.Com every customer can visit a large number of movies by navigating through four movie categories. These four movie categories are: social, action, thriller and comedy movies. All customers have their own personal shopping cart. If a customer intends to buy a movie she/he must simply move the movie into her/his cart by pressing the specific button. Users also have the ability to remove one or more movies from their cart by choosing to delete them. After deciding which movies to buy a customer can easily purchase them by pressing the button "buy".

All navigational moves of a customer are recorded by the system in the statistics database. In this way Vision.Com saves statistics considering the visits in the different categories of movies and movies individually. The same type of statistics is saved for every customer and every movie that is moved to the buyers' cart. The same task is carried out for the movies that are eventually bought by every customer. All of these statistical results are moderated from one to zero and saved in the statistics database.

In particular, Vision.Com interprets users' actions in a way that results in the calculation of users' interests in individual movies and movie categories. Each user's action contributes to the individual user profile by implying degrees of interest into one or another movie category or individual movie. For example, the visit of a user into a movie shows interest of this user to the particular movie and its category. If the user puts this movie into the shopping cart this shows more interest in the particular movie and its category. If the user buys this movie then this shows even more interest whereas if the user takes it out of the shopping cart before payment then there is not any increase in the interest counter. Apart from movie categories that are already presented, other movie features that are taken into consideration by Vision.Com are the following: price range, leading actor and director. The price of every movie belongs to one of the five price ranges: 20 to 25 €, 26 to 30 €, 31 to 35 €, 36 to 40 € and over 41 €.

As a consequence, every customer's interest in one of the above features is recorded as a percentage of his/her visits in movie-pages. For example, the degree of interest of the customer in a particular movie category is calculated by the equation (1).

$$\begin{aligned}
 \text{Interest}_{in_movie_category} = & \\
 = & \frac{\text{Visits}_{in_Specific_Category}}{\text{Visits}_{in_All_Categories}} + \\
 & \frac{\text{Movies}_{moved_to_cart_from_this_category}}{\text{All_movies}_{moved_to_cart}} + \\
 & + \frac{\text{Bought_Movies}_{from_this_category}}{\text{All_bought_movies}}
 \end{aligned} \tag{1}$$

In this way the provided tables of statistics can be easily combined with statistical methods. Moreover, Vision.Com uses an animated agent to inform and help the users throughout the system. The animated agent can be seen at the top-left of the screenshot illustrated in Figure 1.

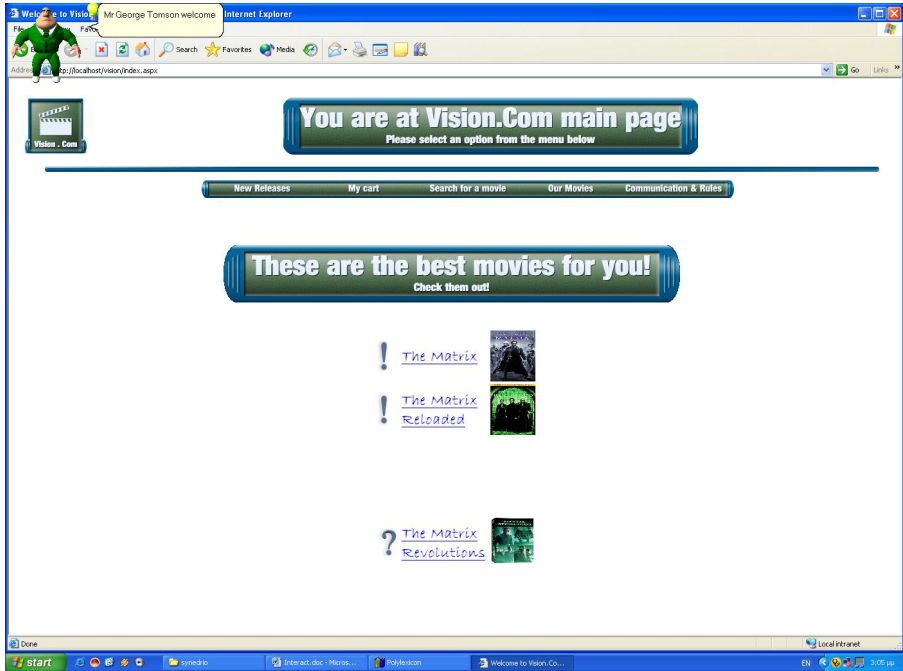


Fig. 1. Screen of Vision.Com movie recommendation web page

3 Data Representation and the AIN Algorithm

In this section we will discuss the data that we used as input to the artificial immune network (AIN) for the construction of users' stereotypes. In order to collect the data an empirical study was conducted. A primary version that did not contain stereotypes of Vision.Com was used by 150 users that bought movies using this particular system. The system collected data about the users' behavior.

The data collected consisted of three parts. Every part is similar to the others. The first one contains statistical data of the visits that every user made to specific movies. The second part contains data of the cart moves (i.e. which movies the user moved into his/her cart). The third part consists of statistical data concerning the preferences on the movies bought by every user. Every record in every part is a vector of the same 80 features that were extracted from the movies' characteristics and represents the references of one user. The 80 features of the vector consist of all the categories, all the sets of prices, all the actors and all the directors that are known to the system. Every constituent in the 80 – dimensional vector is the mean percentage of the user's interest in this particular feature according to Equation 1.

We used these final 80 - dimensional vectors as an input to an AIN algorithm that was derived from the artificial immune systems theory. This was done so as to create representative antibodies that corresponded to the initial data and eventually create stereotype groups by using these antibodies. In the context of Vision.Com, antibodies correspond to neighboring user profiles that are represented by the feature vectors. Specifically, the immune network system clustered users' interests as well as movies and represented each resulting cluster with corresponding antibodies.

The neighborhoods that are produced by the AIN algorithm are utilized as the basis for the construction of a double stereotype that consists of user and movie stereotypes. The stereotypes created, as we will explain in a subsequent section, helped in the logical explanation of the results and led to the creation of a new more accurate and effective user modeling mechanism. In Fig. 2 we demonstrate the results of the application of the AIN algorithm to the data collected during the experimental study. In fact, this process resulted into the identification of 22 representative antibodies that were clustered in a hierarchical tree as illustrated in Fig. 2.

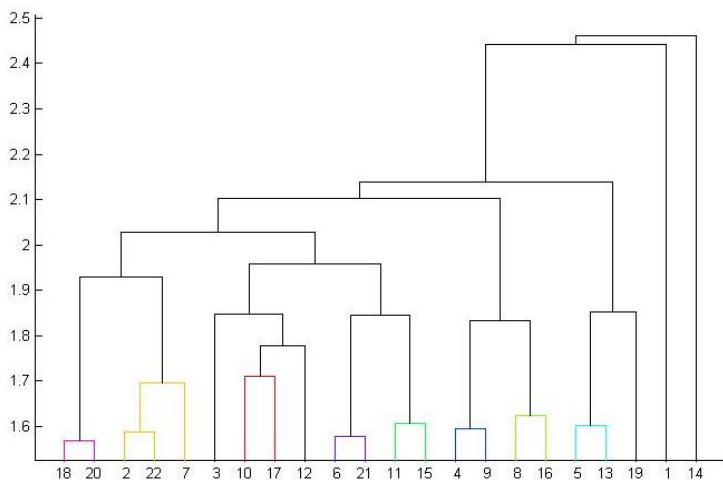


Fig. 2. Tree diagram of clustering the 22 representative antibodies of 150 users of Vision.Com

These antibodies are significant vectors of the same set of features as with the mean vectors. The antibodies present the propagation in the 150 mean vectors that were used as an input to the algorithm. Because every mean vector corresponds to a user, we can also see every antibody as a user that represents preferences of a group of users from the original set. In order to see clearer results we clustered these 22 antibodies into 2, 3, 4, 5 and 6 classes accordingly. By clustering these antibodies into the above classes we observed more clearly the differences between the different preferences in the same feature. The above classes can be seen as stereotypes that can be used for the initialization of a user.

4 Constructing Double Stereotypes Based on the Immune System

After constructing the classes, we used them to build stereotypes. The double classification (users’ interests – movies) was performed in a hierarchical way that resulted in several levels of user stereotypes: At first, there was a coarse classification of two stereotypes which was next refined several times to produce a final classification of six stereotypes as can be seen in Figure 3. Figure 3 illustrates a brief diagram of the hierarchical classification of stereotypes. These stereotypes that have been constructed in the first phase, are then used dynamically by the e-commerce application to infer users’ interests in movies based on a small set of observed users’ actions. In fact, for a new system user, modeling is performed based on the first classification of users. Then, incrementally and while the user interacts with the system, inferences about his/her preferences are drawn from more refined user stereotypes of subsequent levels.

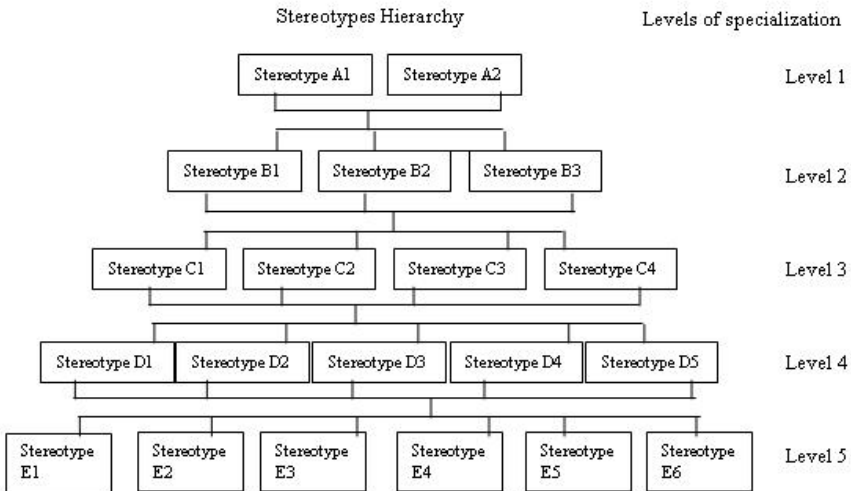


Fig. 3. Hierarchical Diagram of Double Stereotypes

More specifically, at first, on level 1, we created two general stereotypes (Stereotype A1 and Stereotype A2) based on the two general classes. The main differences between these two classes can be observed mainly in the interest in movie categories. In the particular classification, the main difference is in the thriller category. Stereotype A1 shows interests divided into the three categories (action, comedy and social). On the other hand, Stereotype A2 shows an assembly of users’ interest into the thriller movie category. In this way the two stereotypes based on these classes focus their differences of preferences in a category of a movie. As we go further down in the top-down presentation the differences in the stereotypes extend to more features of interest. More specifically, in the stereotypes of the second level of the immune system, differences can be seen not only on the movie categories but on

the interest in medium and high priced movies feature as well. As we proceed to higher levels, stereotypes become more complex and the differences between them extend to more features of interest. At level three of specialization, differences in stereotypes are extended to leading actors as well. At level four, stereotype differences include directors such as Steven Spielberg, Brian de Palma and others. These directors are the ones that attracted the highest interest of users in the previous levels. The leaves of this hierarchical stereotype tree, at level 5, consist of six stereotypes. The differences between them extend to all interest features.

In view of the above results, we have built the hierarchy of stereotypes not only for customers but for movies as well. Whether a customer or a movie belongs or not to a stereotype is measured by the Euclidean distance. The measurement process is opposite to the top-down presentation that we showed just above. If a customer's distance is very far from the leaves then we proceed to the next level above. If in this level the distance is also far we move up to the next. This process is continued until we reach the top level or a threshold of distance that is considered close enough. When this distance is reached the system follows the same process for movies. The movies that are close enough to the customer's stereotype just measured are recommended to the customer through an adaptive hypermedia [2] procedure.

5 Incremental Initialization of User Model Based on Double Stereotypes

Stereotypes have been widely considered as a very effective technique for initializing user models of new users. The answers to the questions concerning which stereotype a user belongs to and which recommendation is best for a particular user, are based on vector subtraction. The decision process is quite complex but here we will present a simpler one because our aim is not to fully explain mathematically this process but to present the system recommendation system. When a new user becomes a member to the system, the e-commerce application creates a profile, sets all interest values into zero (the system assumes that initially the user has no interest in any movie) and starts to monitor his/her actions. After the interaction of the new user with the system (visiting few movies) the system classifies the new user into a stereotype of the first level of specialization. The first level generally considers users' interest concerning the four movie categories. For example, if the new user shows a preference towards thriller movies then s/he classified to Stereotype A2 of the first level, as the main difference between the two stereotypes lies in this movie category. Otherwise if the new user chooses movies from one or more of the other three categories the system classifies the new user into the group that belongs to Stereotype A1 of the first level. In the same way, the system chooses from movie stereotypes the movies that belong to the corresponding movie stereotype. If the user belongs to the first stereotype the system chooses to propose movies from any of the three categories except thrillers in a presentation percentage similar to the interest in every category. The movies that are most close to this movie stereotype are those recommended by the system up to this point of interaction.

As the user continues with moving movies into his/her cart and buying some of them the system moves to the next level of classification that extends stereotypical information to the price features. This is so because in this level stereotypes differ greatly in the price ranges along with movie categories. For example, if the new user

selects movies with medium prices the system chooses to classify him/her to the stereotype that has the greatest interest in this price range concerning always the interest in movie categories. On the other hand, if the new user chooses many movies with high price then the system chooses to classify him/her in a different stereotype always taking in mind the movie categories interest. In this way if two users have similar interests in movie categories and have little differences in price ranges the system will propose similar movie titles to them. Otherwise, if the difference in price ranges is high then the system will most certainly change the most recommended movies. Level four and five of specialization extend the features of interests into interests in leading actors and directors accordingly. In this way, as users show with their actions which actor or director they prefer the system easily classifies them into the respective stereotypes and selects the right movie stereotypes in order to make recommendations. Again if their differences in interest in the actor and directors are low the system chooses to group users in same stereotypes thus emulating the grouping process into the previous level of specialization. On the other hand if these differences are high the users are grouped into different stereotypes of these levels.

The initialization process is conducted until the user reaches level six of specialization. This level represents the leaves in the hierarchical tree of stereotypes and extends the differences in all the features of interests. When a user reaches this amount of interaction with the system s/he has given a lot of information to the system and s/he has been classified in a way that the system knows almost every interest about the user. When the system reaches this level of specialization, not only recommends movies in the appropriate web page but with the use of adaptive hypermedia [2] predicts the next moves of the user and changes the interface dynamically. The specialization in these level is very high and even the smallest difference in user's interest can classify him/her to a different stereotype of this specific level.

6 Conclusions

This paper has presented a user modeling approach that is based on user-stereotypes for an e-commerce application. Our approach has focused on automating the process of the construction of user stereotypes based on automatic observation and analysis of the interaction of e-shop users. The approach that we have presented is novel because it is based on the AIN-algorithm that has not been used for user modeling purposes in other recommender applications in the relevant literature. Moreover, in our approach, evidence on user preferences is collected implicitly without interrupting the users. As a result, the benefit that we have gained is that our system can provide recommendation to all kinds of users (even new ones) on all kinds of movies (even those that have not been seen by any user yet).

References

1. Ardissono L, Torasso P: Dynamic User Modeling in a Web Store Shell, Proceedings of the 14th Conference ECAI, Berlin, Germany, (2000) 621-625.
2. Brusilovsky, P.: Adaptive Hypermedia, User Modeling and User-Adapted Interaction 11. Kluwer Academic Publishers (2001) 87-110

3. Cayzer S., Aickelin U.: A Recommender System based on the Immune Network. Proceedings of the Congress on Evolutionary Computation (2002)
4. De Castro L.,N., Timmis J.: Artificial Immune Systems: A New Computational Intelligence Approach, Edition.1st, Springer-Verlag, London Berlin Heidelberg (2002)
5. Kay J: Stereotypes, Student Models and Scrutability, ITS, LNCS, Springer-Verlag Berlin Heidelberg (2000) 19-30
6. Morrison T., Aickelin U.: An Artificial Immune System as a Recommender for Web Sites. Proceedings of the 1st Conference on ARTificial Immune Systems (ICARIS-2002) Canterbury UK (2002) 161-169
7. Pierrakos D, Paliouras G, Papatheodorou C, Spyropoulos D. C.: Web Usage Mining as a Tool for Personalization: A Survey, Journal of User Modeling and User-Adapted Interaction Vol. 13, Kluwer Academic Publishers, (2003) 311-372
8. Rich E: User Modeling via Stereotypes, Journal of Cognitive Science, Vol. 3, (1979) 329-354
9. Schwarzkopf E.: 2001, An adaptive web site for the UM2001 conference, Proceedings of the UM2001 Workshop on Machine Learning for User Modeling (2001) 77-86.
10. Theodoridis S, Koutroumbas K.: Journal of Pattern Recognition, Edition 3rd, Academic Press, San Diego (2006)

Author Index

- Acevedo, Javier II-238
Adams, Rod I-822
Ahmed, Bilal II-300
Al-Jumeily, Dhiya II-123
Alimi, Adel M. I-240
Allahverdi, Novruz II-467
Allende, Héctor II-355, II-554
Altun, Adem Alpaslan II-467
Álvarez, José L. I-39
Amari, Shun-ichi II-271
Andrejková, Gabriela I-404
Anisetti, Marco I-684
Arslan, Ahmet I-694
- Bae, Hyeon-Deok II-534
Baer, Philipp A. I-167
Bağ, Michał II-133
Barreira, Noelia I-202
Barrón, Ricardo II-55
Becerra, Carlos II-554
Beliczynski, Bartłomiej II-46
Bellandi, Valerio I-684
Ben Amor, Heni II-641
Biçici, Ergun I-739
Bielecki, Andrzej II-133
Bielski, Conrad II-500
Bilgin, Mehmet Zeki II-713
Borkowski, Adam I-102
Bravo, Crescencio I-649
Browne, Jim I-498
Byun, Yeun-Sub I-657
- Cai, Nian II-582
Chacón, Máx II-355
Chai, Zhilei I-376, I-394
Chakraborty, Uday K. I-77
Chandran, Harish I-11
Chang, Ping-Teng I-631
Charuwat, Thitipong I-230
Chen, Ping-Jie II-246
Chen, Ta-Cheng I-526
Cheng, Jao-Hong I-614
Chiu, Deng-Yiv II-246
Chiu, Singa Wang II-525
Chiu, Yuh-Wen I-614
- Cho, Dal-ho II-440
Cho, Seong Jin II-300
Choi, Dae-Young I-517
Choi, Hun II-534
Choi, Jae-Seung II-153
Choi, Jeoung-Nae I-622
Choi, Se-Hyu I-306
Choi, Seungjin II-271
Choi, YoungSik I-588
Chong, Kil To II-676
Chongchao, Huang I-341
Choraś, Michał II-407, II-424
Choraś, Ryszard S. II-407
Chou, Chih-Hsun I-604
Chung, Chung-Yu II-525
Chung, Gyo-Bum II-738
Chung, Yongwha II-432
Cichocki, Andrzej II-271, II-373
Cieslik, Dominik II-624
Couchet, Jorge I-730
Cutello, Vincenzo I-93
Cyganek, Bogusław II-508
Czajewski, Witold II-633
- Dąbrowski, Paweł I-194
Davey, Neil I-822
Deguchi, Toshinori II-37
Deißler, Tobias II-616
Dereli, Türkay I-508
Dioşan, Laura II-218
Dobnikar, Andrej II-63
Dominik, Andrzej I-772
Dorado, Julián I-276
Dreżewski, Rafał I-67
Du, Wei I-296
Duque, Rafael I-649
Dvořák, Jakub I-721
Dzemyda, Gintautas II-179, II-544
Dzielinski, Andrzej I-414
- Ebert, Alfons II-492, II-616
Erig Lima, Carlos R. I-159
- Fabijański, Paweł I-640
Faling, Gui I-341

- Fang, Wei I-376
 Ferreira, Enrique I-730
 Folan, Paul I-498
 Fonseca, André I-730
 Fraser, Robert I-758

 Gabrijel, Ivan II-63
 Galán-Marín, Gloria I-461, II-98
 Galbiati, Jorge II-554
 Gambin, Anna I-422
 Gámez, José A. I-806
 Gammoudi, Jamil I-148
 García, Cristina II-667
 Gegúndez, Manuel E. I-39
 Geihs, Kurt I-167
 Ghazali, Rozaida II-123
 Gil, Pedro II-238
 Glasgow, Janice I-758
 Godoy Jr., Walter I-358
 Gómez-Tato, Andrés II-208
 González, Jesus I-85
 González-Castaño, Francisco J. II-208
 González de-la-Rosa, Juan-José I-782
 Górriz, Juan-Manuel I-782
 Grześ, Marek I-1
 Guillén, Alberto I-85
 Güneş, Salih II-338
 Guo, Xinchun II-189
 Güven, Aysegül II-338
 Gwun, Ou-Bong II-590

 Hahn, Minsoo II-382
 Hamdani, Tarek M. I-240
 Han, Changwook I-257
 Han, Liyan I-314
 Han, Song-yi II-440
 Han, Soowhan I-257
 Hembecker, Fernanda I-358
 Herrera, Luis J. I-85
 Hlaváčková-Schindler, Kateřina I-790
 Hsieh, Yi-Chih I-526
 Huang, Hai I-314
 Hussain, Abir Jaafar II-123
 Hwang, Hyung-Soo I-622

 Ibáñez, Óscar I-202
 Ikemoto, Shuhei II-641
 Im, Dae-Yeong II-730
 İnal, Melih I-266
 Ishiguro, Hiroshi II-641

 Ishii, Naohiro II-37
 Ivanikovas, Sergejus II-179
 Iwanowski, Marcin II-606

 Janežič, Dušanka II-399
 Jang, Seong-Whan I-666
 Jarur, Mary Carmen II-107
 Jedruch, Wojciech I-386
 Jedrzejowicz, Piotr I-480
 Jeon, Gwanggil I-684
 Jeong, Dae Sik II-415
 Jeong, Jechang I-684
 Jeong, Jong-Cheol II-364
 Jeong, SangBae II-382
 Jiang, Jingqing I-562
 Jo, Geun-Sik II-71
 Joseph, Shaine I-77
 Jung, Bernhard II-641
 Jung, Jin-Guk II-71
 Jung, Seunghwan II-432

 Kacalak, Wojciech I-596
 Kaczorek, Tadeusz II-694
 Kainen, Paul C. II-11
 Kang, Byung Jun II-415
 Kang, Hyung W. I-77
 Kara, Sadık II-338
 Karci, Ali I-450
 Karray, Fakhri I-240
 Kawaguchi, Masashi II-37
 Khan, Mohammed A.U. II-300
 Kim, Eun-Mi II-364
 Kim, Gwang-Ha II-290
 Kim, Hyongsuk II-346
 Kim, Hyun-Ki I-666
 Kim, Il-hwan II-722
 Kim, Jinhwa I-830
 Kim, Kwang-Baek II-290, II-572
 Kim, Mi-Young I-814
 Kim, Min-Soo I-657
 Kim, Minhwan II-572
 Kim, Sang-Chul II-659
 Kim, Sungshin II-290
 Kim, Tae-Seong II-300
 Kim, Young-Chul II-676
 Kisiel-Dorohinicki, Marek I-138
 Kleiber, Michał I-570
 Koh, Eun Jin II-517
 Kokosiński, Zbigniew I-211
 Konc, Janez II-399

- Kong, Jun II-309
 Koperwas, Jakub I-702
 Köppen, Mario I-323
 Korbicz, Józef II-19
 Kozak, Karol II-327
 Kozak, Marta II-327
 Kozłowski, Bartosz I-49
 Krasnogor, Natalio I-93
 Krętoski, Marek I-1
 Kulworawanichpong, Thanatchai I-230
 Kumeresh, Arjun I-11
 Kurasova, Olga II-544
 Kůrková, Věra II-11
 Kupis, Paweł I-23
 Kusiak, Magdalena I-432
 Kwarciany, Krzysztof I-211
 Kwolek, Bogdan II-599
- Lafuente, Sergio II-238
 Lagoda, Ryszard I-640
 Lai, Choi-Hong I-394
 Lai, Kin Keung II-262
 Lawryńczuk, Maciej II-143
 Lee, Bae-Ho II-364
 Lee, Chong Ho I-286
 Lee, Dae-Young II-534
 Lee, Eui Chul II-415
 Lee, Imgeun I-257
 Lee, In-Tae I-666
 Lee, Inbok I-554
 Lee, Jae-kang II-722
 Lee, Jee-hyong I-441
 Lee, Ji-Yeoun II-382
 Lee, Ju-Sang II-730
 Lee, Keon-myung I-441
 Lee, Kidong I-830
 Lee, Sangyoung II-440
 Lee, Sungju II-432
 Lee, Sungyoung II-300
 Lee, Yung-Cheng I-526
 Lengeňová, Helena I-404
 Lévano, Marcos II-355
 Li, Hongzhi II-309
 Li, Ming II-582
 Li, Qing I-498
 Liang, Yanchun I-296, I-562
 Liberski, Paweł II-346
 Lin, Hong-Dar II-525
 Lin, Kuo-Ping I-631
 Lipiński, Dariusz I-596
- Lipinski, Piotr II-391
 Liu, Fan-Yong I-674
 Lloret, Isidro I-782
 Lopatka, Rafal I-414
 Lopes, Heitor S. I-159, I-358
 López-Rodríguez, Domingo I-461, II-98
 Lotfi, Naser I-110
 Lotfi, Shahriar I-110
 Lotrič, Uroš II-254
 Lu, Yinghua II-309
 Luo, Zhi-Jie I-604
- Maddouri, Mondher I-148
 Maldonado, Saturnino II-238
 Mańdziuk, Jacek I-23, I-432
 Manrique, Daniel I-730
 Mariańska, Bożena II-318
 Markiewicz, Tomasz II-318
 Martínez-Álvarez, Rafael P. II-208
 Masouris, Michael II-457
 Mateo, Juan L. I-806
 Mati, Michal I-404
 Medvedev, Viktor II-179
 Merabti, Madjid II-123
 Mérida-Casermeiro, Enrique I-461,
 II-98
 Middelmann, Wolfgang II-492, II-616
 Minato, Takashi II-641
 Mohamed Ben Ali, Yamina I-128
 Moon, Daesung II-432
 Mora, Marco II-107
 Moreno, José Alí II-667
 Mroczkowski, Piotr II-424
 Mrugalski, Marcin II-19
 Muñoz, Antonio Moreno I-782
- Nam, Mi Young II-517
 Nicosia, Giuseppe I-93
 Nikodem, Piotr I-102
 Noh, JiSung I-588
 Nowak, Hans II-355
- Ogiela, Marek R. II-477
 Ogryczak, Włodzimierz I-578
 Oh, Sung-Kwun I-622, I-666
 Oh, Tae-seok II-722
 Oltean, Mihai I-220, II-218
 Orłowska, Maria I-536
 Ortiz-de-Lazcano-Lobato, Juan M.
 I-461, II-98

- Osowski, Stanisław II-318, II-373
 Ospina, Juan I-120
 Özcan, Ender I-366

 Pae, Ho-Young II-364
 Paechter, Ben I-85
 Pai, Ping-Feng I-631
 Palacios, Pablo I-39
 Pang, Yunjie I-546
 Park, Doo-kyung I-441
 Park, Hyun-Ae II-440
 Park, Jang-Hyun II-730
 Park, Jong-Ho II-676
 Park, Junghee I-830
 Park, Kang Ryoung II-415, II-440
 Park, Kyo-hyun I-441
 Park, Seung-Jin II-153
 Parsa, Saeed I-110
 Pavone, Mario I-93
 Pazos, Alejandro I-276
 Pearson, David W. I-767
 Pecuchet, Jean-Pierre II-218
 Pempera, Jaroslaw I-194
 Penedo, Manuel G. I-202
 Pettersson, Frank II-115
 Piao, Chang Hao I-286
 Pierluissi, Luis I-31
 Pinto, Pedro C. I-350
 Plemmons, Robert II-271
 Podolak, Igor T. I-749
 Polat, Kemal II-338
 Polat, Övünç II-161
 Pomares, Hector I-85
 Prodan, Lucian I-174
 Puangdownreong, Deacha II-747
 Puerta, José M. I-806
 Puntonet, Carlos G. I-782

 Qi, Miao II-309

 Rabuñal, Juan I-276
 Rasheed, Tahir II-300
 Ratajczak-Ropel, Ewa I-480
 Raudys, Sarunas I-711, II-1
 Rhee, Phill Kyu II-517
 Ribeiro, Bernardete II-199, II-228
 Rivero, Daniel I-276
 Robinson, Mark I-822
 Rocco S., Claudio M. I-31
 Rodríguez-Hernández, Pedro S. II-208

 Rogozan, Alexandrina II-218
 Rojas, Ignacio I-85
 Romaszkiwicz, Andrzej I-578
 Rosas, Lorna V. II-564
 Rudnicki, Marek II-281
 Ruican, Cristian I-174
 Runkler, Thomas A. I-350
 Rust, Alistair G. I-822
 Ryoo, Young-Jae II-685, II-730
 Rysz, Andrzej II-373

 Sadiq, Shazia I-536
 Şakiroğlu, Ayşe Merve I-694
 Sandou, Guillaume I-332
 Sanguineti, Marcello II-11
 Santos, José I-202
 Savický, Petr I-721
 Savvopoulos, Anastasios I-837
 Saxén, Henrik II-115
 Sbarbaro, Daniel II-107
 Sena Daş, Gülesin I-508
 Serban, Ana-Talida I-332
 Sharabi, Offer I-822
 Shin, Yun-su II-722
 Shukla, Pradyumn Kumar I-58
 Siegmann, Philip II-238
 Sienkiewicz, Rafal I-386
 Silva, Catarina II-228
 Silva, Rafael R. I-159
 Siwik, Leszek I-67, I-138
 Skoneczny, Slawomir II-624
 Smith, Darren I-488
 Smutnicki, Czeslaw I-194
 Sohn, Sang-Wook II-534
 Soille, Pierre II-500, II-606
 Son, Eun-Ho II-676
 Song, Ha Yoon I-554
 Song, Ju-Whan II-590
 Sossa, Humberto II-55
 Sotiropoulos, Dionisos N. I-837
 Sousa, João M.C. I-350
 Staniak, Maciej II-633
 Stapor, Katarzyna II-327
 Stasiak, Bartłomiej II-27
 Stathopoulou, Ioanna-Ourania II-449
 Šter, Branko II-63
 Strumiłło, Paweł II-281
 Strzelecki, Michal II-346
 Suh, Jae-Won II-534
 Sujitjorn, Sarawut II-747

- Sun, Fangxun I-296
 Sun, Jun I-376, I-394
 Sun, Yi I-822
 Sung, Andrew H. II-228
 Świdorski, Bartosz II-373
 Szatkiewicz, Tomasz II-80
 Szczurek, Ewa I-422
- Tadeusiewicz, Ryszard II-477
 Tambouratzis, Tatiana II-169,
 II-457, II-649
 te Boekhorst, Rene I-822
 Thoennesen, Ulrich II-492, II-616
 Tian, Lei I-314
 Tiesong, Hu I-341
 Trebar, Mira II-254
 Trojanowski, Krzysztof I-184
 Tsihrintzis, George A. II-449, I-837
- Uddin, Mohammed Nazim II-71
 Udrescu, Mihai I-174
 Unold, Olgierd I-798
- Vargas, Héctor II-564
 Vázquez, Roberto A. II-55
 Vejmelka, Martin I-790
 Vélez, Mario I-120
 Venkateswaran, Nagarajan I-11
 Viet, Nguyen Hoang I-570
 Virvou, Maria I-837
 Vladutiu, Mircea I-174
- Walczak, Krzysztof I-702
 Walczak, Zbigniew I-772
 Wałędzik, Karol I-432
 Waller, Matias II-115
 Wang, Chaoyong II-189
 Wang, Hui II-582
 Wang, Jin I-286
 Wang, Qing I-498
 Wang, Rujuan II-309
 Wang, Shouyang II-262
 Wang, Shuqin I-296
 Wang, Xin I-546
 Wang, Xiumei I-296
- Wang, Yan I-296
 Wang, Yijing II-88, II-704
 Wang, Yunxiao I-546
 Wang, Zhengxuan I-546
 Wawrzyński, Paweł I-470
 Webb, Barbara I-488
 Weise, Thomas I-167
 Wessnitzer, Jan I-488
 Wojciechowski, Jacek I-772
 Won, Jin-Myung I-240
 Woo, Young Woon II-572
 Wu, Chunguo I-562, II-189
 Wysocka-Schillak, Felicja I-248
- Xianing, Wu I-341
 Xiao, TianYuan I-498
 Xu, Wenbo I-376, I-394
- Yan, Jiang I-341
 Yang, Hyong-Yeol II-730
 Yang, Jie II-582
 Yang, Jinhui I-562
 Yang, Li-An I-604
 Yatsymirskyy, Mykhaylo II-27, II-391
 Yeh, Chung-Hsing I-614
 Yıldırım, Tülay II-161
 Yılmaz, Murat I-366
 Yoon, Tae-bok I-441
 Yoon, Tae-jun II-722
 Yoshida, Kaori I-323
 Yu, Kun-Ming I-604
 Yu, Lean II-262
 Yu, Xiao I-546
 Yuan, Bo I-536
 Yuret, Deniz I-739
- Zalewska, Anna II-346
 Zanella, Vittorio II-564
 Zdunek, Rafal II-271
 Zhang, Na I-562
 Zhang, Wenbo II-189
 Zhou, Chunguang I-296
 Zhou, Jian I-498
 Zhou, Jiayi I-604
 Zuo, Zhiqiang II-88, II-704