ADVANCES IN *SOFT COMPUTING*

Ajith Abraham · Bernard de Baets
Mario Köppen · Bertram Nickolay
Editors

Applied Soft Computing Techniques: The Challenge of Complexity

Springer

Ajith Abraham, Bernard de Baets, Mario Köppen, Bertram Nickolay (Eds.)

Applied Soft Computing Technologies: The Challenge of Complexity

Advances in Soft Computing

Further volumes of this series
can be found on our homepage:
springer.com

Mieczysław Kłopotek, Maciej Michalewicz and Sławomir T. Wierzchoń (Eds.)
Intelligent Information Systems 2002, 2002
ISBN 3-7908-1509-8

Andrea Bonarini, Francesco Masulli and Gabriella Pasi (Eds.)
Soft Computing Applications, 2002
ISBN 3-7908-1544-6

Leszek Rutkowski, Janusz Kacprzyk (Eds.)
Neural Networks and Soft Computing, 2003
ISBN 3-7908-0005-8

Jürgen Franke, Gholamreza Nakhaeizadeh, Ingrid Renz (Eds.)
Text Mining, 2003
ISBN 3-7908-0041-4

Tetsuzo Tanino, Tamaki Tanaka, Masahiro Inuiguchi
Multi-Objective Programming and Goal Programming, 2003
ISBN 3-540-00653-2

Mieczysław Kłopotek, Sławomir T. Wierzchoń, Krzysztof Trojanowski (Eds.)
Intelligent Information Processing and Web Mining, 2003
ISBN 3-540-00843-8

Ajith Abraham, Katrin Franke, Mario Köppen (Eds.)
Intelligent Systems Design and Applications, 2003
ISBN 3-540-40426-0

Ahmad Lotfi, Jonathan M. Garibaldi (Eds.)
Applications and Science in Soft-Computing, 2004
ISBN 3-540-40856-8

Mieczysław Kłopotek, Sławomir T. Wierzchoń, Krzysztof Trojanowski (Eds.)
Intelligent Information Processing and Web Mining, 2004
ISBN 3-540-21331-7

Miguel López-Díaz, María ç. Gil, Przemysław Grzegorzewski, Olgierd Hryniewicz, Jonathan Lawry
Soft Methodology and Random Information Systems, 2004
ISBN 3-540-22264-2

Kwang H. Lee
First Course on Fuzzy Theory and Applications, 2005
ISBN 3-540-22988-4

Barbara Dunin-Keplicz, Andrzej Jankowski, Andrzej Skowron, Marcin Szczuka
Monitoring, Security, and Rescue Techniques in Multiagent Systems, 2005
ISBN 3-540-23245-1

Bernd Reusch (Ed.)
Computational Intelligence, Theory and Applications: International Conference 8th Fuzzy Days in Dortmund, Germany, Sept. 29 – Oct. 01, 2004 Proceedings, 2005
ISBN 3-540-2280-1

Frank Hoffmann, Mario Köppen, Frank Klawonn, Rajkumar Roy (Eds.)
Soft Computing: Methodologies and Applications, 2005
ISBN 3-540-25726-8

Ajith Abraham, Bernard de Baets, Mario Köppen, Bertram Nickolay (Eds.)
Applied Soft Computing Technologies: The Challenge of Complexity, 2006
ISBN 3-540-31649-3

Ajith Abraham
Bernard de Baets
Mario Köppen
Bertram Nickolay
(Eds.)

Applied Soft Computing Technologies: The Challenge of Complexity

Springer

Ajith Abraham
School of Computer Science and Engineering
Chung-Ang University
Heukseok-dong 221
156-756 Seoul, Korea
E-mail: ajith.abraham@ieee.org

Mario Köppen
Fraunhofer IPK Berlin
Dept. Automation Technologies
Pascalstr. 8-9
10587 Berlin, Germany
E-mail:mario.koeppen@ipk.fraunhofer.de

Bernard de Baets
Department of Applied Mathematics
Biometrics and Process Control
University Gent
Coupure Links 653
9000 Gent, Belgium
E-mail: bernard.debaets@rug.ac.be

Bertram Nickolay
Fraunhofer IPK Berlin
Dept. Automation Technologies
Pascalstr. 8-9
10587 Berlin, Germany
E-mail:bertram.nickolay@ipk.fraunhofer.de

Library of Congress Control Number: 2005938949

ISSN print edition: 1615-3871
ISSN electronic edition: 1860-0794
ISBN-10 3-540-31649-3 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-31649-7 Springer Berlin Heidelberg New York

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

Typesetting: by the authors and TechBooks using a Springer LaTeX macro package

Printed on acid-free paper SPIN: 11376354 89/TechBooks 5 4 3 2 1 0

WSC9 – Honorary Chair's Message

Hello and welcome to the 9th Online World Conference on Soft Computing in Industrial Applications (WSC9). This conference, brought to you from everywhere in the world at once, provides an opportunity for researchers and practitioners to meet in a virtual environment to present their latest results and to exchange ideas. With this conference we are taking advantage of the latest advances in hard computing to discuss the newest ideas in soft computing. There is no doubt that with this type of online conferences we are pioneering into the future. The ability to transport yourself anywhere in the world at the speed of light, in the comfort and safety of your own home or office, can't be seen as anything but amazing. If there are participants from another galaxy, please let us know!!

Soft computing, a name introduced by LA. Zadeh, can be seen as a confluence of a number of related, complementary and supporting disciplines. Among the central components are neural networks, fuzzy logic and genetic algorithms. Rough sets, probabilistic reasoning and significant portions of artificial intelligence also come under the umbrella of soft computing. The field of computational intelligence is closely related. Central to the field of soft computing is an attempt to provide tools that can enable systems to attain a level of performance similar to human beings. Among other things, this requires an ability to represent and manipulate imprecise and granular information.

While the areas of potential applications of soft computing is virtually unbounded, I shall point out, with a high degree of bias, some areas in which I believe the possible applications of soft computing are particularly interesting. Clearly the Internet is one such area. As we move from the initial "Gee Whiz" phase to the next stage where success requires intelligent manipulation of knowledge and data, website architects must draw upon tools from soft computing. As a particular example, the development of intelligent agents for retrieving and searching for information will benefit from application of these technologies. Fuzzy logic will play an important role in the development of the semantic web. True customization and personalization of websites must take advantage of soft computing. The related area of data mining is already using many ideas from this field. Many people believe the coming century will see many benefits from recent discoveries in microbiology, genetics and proteinomics. Here soft computing can help, especially in the development of algorithms needed for partial matching, clustering and pattern recognition.

I would like to thank the organizers of this conference, particularly the General Chair Ajith Abraham and the Program Chair Mario Köppen for their efforts in making this conference happen.

I now declare the 9th Online Conference on Soft Computing in Industrial Applications open. I look forward to a lively exchange of ideas.

Ronald R. Yager
New York, USA

WFSC Chairperson's Message

As the Chairperson of World Federation of Soft Computing (WFSC) it is my great pleasure to welcome all participants of the 9th On-line World Conference on Soft Computing in Industrial Applications (WSC9) in the Cyberspace. The conference is the ninth of WSC series sponsored by WFSC.

Since we started the cyber-conference in 1996, this series has been expanded and received wider support from the researchers in the field of soft computing. The conference has been recognised internationally IEEE SMC Society, NAFIPS, IFSA and EUSFLAT. The conferences have removed the financial burden from the authors and participants, and thus have attracted good quality papers from all over the world. The registered participants have also represented academia and industries from many countries around the world. The conferences have established a new way of publishing technical papers and discussing papers in greater detail. Every year we try to introduce innovation in the way we manage the conference. In future we are looking for feedback from our participants to improve the quality and branding of the conference.

The 'World Federation of Soft Computing (WFSC)', which is a virtual society of active researchers in the field, has been established since 1999. The aim of the Federation is to promote 'Soft Computing' across the world. WFSC also promotes other organisations interested in Soft Computing. The Federation also has an official Journal called 'Applied Soft Computing (ASOC)', published by Elsevier. Best Papers application papers from WSC9 will be invited to submit revised and extended versions to the ASOC Journal on a FAST TRACK basis. The papers will be published in Sciencedirect website soon after they are accepted, which enables authors to publish their work fast and readers get the latest work in Soft Computing on their desktop. A free hardcopy of the volume is provided to the authors by the end of the year.

I am looking forward to meet you all during the conference! Let's make an effort to start a good debate on Soft Computing research and its application.

Rajkumar Roy
Chair of World Federation of Soft Computing (WFSC)

WSC9 Chair's Welcome Message

On behalf of the 9th *Online Conference on Soft Computing in Industrial Applications* (WSC9) organizing committee, we wish to extend a very warm welcome to this exciting meeting. The conference program committee has organized an exciting and invigorating program comprising of presentations from distinguished experts in the field and important and wide-ranging contributions on state-of-the-art research that provide new insights into the '*Current Innovations in Soft Computing Applications'*. All registered participants had the opportunity to attend the various sessions (online papers, presentations and demonstrations) from September 20 to October 08, 2004. For online interaction, several discussion boards were also setup and moderated by the respective session chairs.

During the past few years, WSC conference has achieved steady growth as evident from the support received from the academia and industry. From the very beginning of WSC9, we have been receiving so much support from several institutions around the globe. Our special thanks go to New Mexico Technology, USA; Oxford University, UK; Ben-Gurion University, Israel; Beijing City University, China; University of South Australia, Australia and Fraunhofer-IPK Berlin, Germany for providing the resources to maintain the web sites of WSC9. WSC9 was hosted by the 'World Federation on Soft Computing' and is technically co-sponsored by IEEE Systems Man and Cybernetics Society, North American Fuzzy Information Processing Society, European Society for Fuzzy Logic and Technology, International Fuzzy Systems Association and Elsevier Science, The Netherlands. Technical submissions were invited in, but not limited to, any of the following areas:

- Fuzzy Control
- Neuro-Fuzzy Systems
- Genetic Fuzzy Systems
- Neuro-Fuzzy-Genetic Systems
- Software Agent Systems and Architectures
- Multi-Agent Architectures and Collaborative Learning
- Ant Colony Optimization
- Fuzzy Information Fusion
- Rough Sets
- Bayesian Networks
- Fuzzy Image Processing
- Bio-inspired Systems
- Artificial Neural Networks
- Support Vector Machines
- Evolutionary Algorithms, Simulated Annealing, Tabu Search
- Evolutionary Multiobjective Optimization
- Hybrid Optimization Techniques (integration of global search and local search techniques)

- Simulation Environments
- Data Mining
- Industrial Applications of Soft Computing
- Intelligent Information Retrieval
- Language Processing
- Robotics
- Autonomous Reasoning
- Fault Diagnosis
- Bioinformatics
- Web Intelligence
- Speech Processing
- Business Information Systems
- Knowledge Management

WSC9 received 6 technical session proposals and over 100 technical papers from 27 countries. These papers focused on a variety of soft computing applications spanning from information security, bioinformatics, manufacturing, scheduling, and so on. Each submission was peer reviewed by three or more program committee members. These accepted papers offer stimulating insights into emerging soft computing technologies and their applications. The papers are grouped into several technical sessions each with a specific application focus. Thanks to Katrin Franke (Fraunhofer IPK Berlin, Germany) for all the wonderful coordination work related to the organization of invited special sessions.

We would like to express our sincere thanks to all the authors and members of the program committee that has made this event a success. We are also grateful to Professors Ronald Yager and Witold Pedrycz for their valuable time in preparing the plenary lectures. Finally, we hope that you will find the presentations and discussions to be a valuable resource in your professional, research, and educational activities whether you are a student, academic, researcher, or a practicing professional. Enjoy!

Ajith Abraham
WSC9 – General Chair

Mario Köppen
WSC9 – Program Chair

WSC9 – Organization

Honorary Chair

Ronald Yager, Iona College, USA

General Chair

Ajith Abraham, IITA Professorship Program,
Chung-Ang University, Seoul, Korea

Program Chair

Mario Köppen, Fraunhofer IPK Berlin, Germany

International Co-chairs

Frank Hoffmann, University of Dortmund, Germany
Avineri Erel, Ben-Gurion University of the Negev, Israel

Sponsorship Chair

Rajkumar Roy, University of Cranfield, UK

Publication Chair

Bernard de Baets, Ghent University, Belgium

Virtual Exhibition Chair

Fabio Zambetta, University of Bari, Italy

Special Events Chair

Katrin Franke, Fraunhofer IPK Berlin, Germany

Industrial Liason Chairs

Antony Satyadas, IBM, USA
Suthikshn Kumar, Larsen & Toubro Infotech, India

Best Paper Award Chair

Janos Abonyi, University of Veszprém, Hungary

International Technical Program Committee

Andew Sung, New Mexico Institute of Mining and Technology, USA
Andras Kornai, Budapest Institute of Technology, Hungary
Andrea Bonarini, Politecnico di Milano, Italy
Andreas König, University of Kaiserslautern, Germany
Andrew Huang, Nanyang Technological University, Singapore
António Gaspar-Cunha, University of Minho, Portugal
Aureli Soria Frisch, FhG IPK Berlin, Catalonia
Avineri Erel, Ben-Gurion University of the Negev, Israel
Bart Baesens, Katholieke Universiteit Leuven, Belgium
Berend Jan van der Zwaag, University of Twente, The Netherlands
Brian Carse, University of the West of England Bristol, UK
Carlos Coello, CINVESTAV/IPN Mexico, Mexico
Carlos Pena, Swiss Federal Institute of Technology Lausanne, Switzerland
Chee Peng, Universiti Sains Malaysia, Malaysia
Christian Wöhler, Daimler Chrysler AG, Germany
Christophe Marsala, University Pierre et Marie Curie Paris, France
Crina Grosan, Babes-Bolyai University, Romania
Daming Shi, Nanyang Technological University, Singapore
Detlef D. Nauck, BTexact Technologies Ipswich, UK
Eduardo Gomez, Universidad La Salle , Mexico
Edward W. Tunstel, Califormia Institute of Technology, Pasadena, USA
Francisco Herrera, E.T.S. Ingeneria Informatica Granada, Spain
Frank Hoffmann, University of Dortmund, Germany
Frank Klawonn, University of Applied Sciences Braunschweig, Germany
Gerardo Rossel, Universidad de Buenos Aires, Argentina
Guiseppe di Fatta, ICAR-CNR, Italy
Guy de Tre, Ghent University, Belgium
Hani Abdel Kader Hagras, University of Essex, UK
Hisao Ishibuchi, Osaka Prefecture University, Japan
Horia-Nicolai, Technical University of Iasi, Romania
Hung T. Nguyen, New Mexico State University, USA
Imran Maqsood, University of Regina , Canada
Ioannis Hatzilygeroudis, University of Patras, Greece
Jae Oh, Syracause University, USA

James Kwok, Hong Kong University of Science and Technology, Hong Kong
Janus Abonyi, University of Veszprem, Hungary
Javier Ruiz-del-Solar, Universidad de Chile Santiago, Chile
Jenny Lewis, Mass Customization and Personalization International, Inc., USA
John Debanham, University of Technology Sydney, Australia
José M. Benitez, University of Granada, Spain
Junbin Gao, University of New England, Australia
K. Balasubramanian, Yahoo! Inc. , India
Katrin Franke, Fraunhofer IPK, Germany
Leandro dos Santos Coelho, Pontificia Universidade Catolica do Paraná Curitib, Brazil
Ling Ling Li, Mass Customization and Personalization International, Inc., USA
Luis Magdalena, Ciudad Universitaria Madrid, Spain
Malik M.A. Khalfan, University of Salford , United Kingdom
Mark Embrechts, Rensselear Polytechnic Institute, USA
Marley Vellasco, University College London, UK
Mauro Dell'Orco, University of Bari, Italy
Muhammad Sarfraz, King Fahd University of Petroleum and Minerals, Saudi Arabia
Nadia Nedjah, State University of Rio de Janeiro, Brazil
Okyay Kaynak, Bogazici University, Turkey
Oscar Castillo, Instituto Tecnológico de Tijuana , Mexico
Oscar Cordon, University of Granada, Spain
Paramasivan Saratchandran, Nanyang Technological University, Singapore
Patricia Melin, Instituto Tecnológico de Tijuana , Mexico
Ravi Jain, University of South Australia, Australia
Richard Regli, Mass Customization and Personalization International, Inc., USA
S.H. Srinivasan, Satyam Computers, India
Sankar Kumar Pal, Indian Statistical Institute Kolkata, India
Senen Barro Ameneiro, SEVIMAV Santiago de Compostela, Spain
Sigal Berman, Weizmann Institute of Science, Israel
Sonja Petrovic-Lazarevic, Monash University, Australia
Soumya Banerjee, Institute of Management Studies (Dehradun), India
Sudhir Barai, Indian Institute of Technology, India
Takeshi Furuhashi, Mie University, Japan
Teresa Lee, Mass Customization and Personalization International, Inc., USA
Thomas Sudcamp, Wright State University, USA
Tomo Hiroyasu, Doshisha University, Japan
Vasile Palade, Oxford University, UK
Vicenc Torra, Universitat Autonoma de Barcelona, Spain
Vitorino Ramos, CRVM IST, Portugal
Walter D. Potter, University of Georgia, USA

WSC9 Technical Sponsors

WSC9
is Organized by
World Federation of Soft Computing
WSC 9 Technical Sponsors
ELSEVIER
Springer
SYSTEMS, MAN, AND CYBERNETICS SOCIETY
IEEE
NAFIPS
EUROPEAN SOCIETY FOR FUZZY LOGIC AND TECHNOLOGY

Contents

Part I Plenary Presentations

Part II Classification and Clustering

Part III Optimization

Part IV Diagnosis and Fault Tolerance

Part V Tracking and Surveillance

Part VI Scheduling and Layout

Part VII Complexity Management

Part VIII Manufacturing and Production

Part IX Signal Processing

Part X Computer Security

Part XI Bioinformatics

Part XII Text Processing

Part XIII Algorithm Design

Part XIV Control

Part XV Hybrid Intelligent Systems using Fuzzy Logic, Neural Networks and Genetic Algorithms

Part XVI Recent Developments in Support Vector and Kernel Machines

Part XVII Robotics

Part XVIII Soft Computing and Hybrid Intelligent Systems in Product Design and Development

List of Contributors

Abbattista, Fabio
Università degli Studi di Bari
Via Orabona 4, 70125 Bari, Italy
fabio@di.uniba.it

Agrawal, Piyush
Department of Civil Engineering
Indian Institute of Technology
Kharagpur
Kharagpur 721 302, India
piyush0101@yahoo.com

Almeida, Otacílio M.
Electrical Engineering Department
Federal University of Ceara - Brazil
Campus Pici - CP 6001
CEP 60455 - 760

Antonsson, Erik K.
Engineering Design Research
Laboratory
California Institute of Technology
Pasadena, California 91125, USA
erik@design.caltech.edu

Azmi-Murad, Masrah
Department of Engineering Mathematics
University of Bristol
Bristol, BS8 1TR, UK
masrah@fsktm.upm.edu.my

Baños, R.
Dept. Arquitectura de Computadores
y Electrónica
University of Almería, Spain
La Cañada de San Urbano s/n,
04120 Almería (Spain)
rbanos@ace.ual.es

Barai, S V
Department of Civil Engineering
Indian Institute of Technology
Kharagpur
Kharagpur 721 302, India
skbarai@civil.iitkgp.ernet.in

Berman, Sigal
Department of Computer Science
and Applied Mathematics
Weizmann Institute of Science
76100, Rehovot, Israel
sigal.berman@weizmann.ac.il

Bessant, Conrad
Department of Analytical Science
and Informatics
Institute of BioScience and IT
Cranfield University, Silsoe Campus
Bedfordshire, UK. MK45 4DT

Bezerra, Luis Daniel S.
Electrical Engineering Department

Federal University of Ceara - Brazil
Campus Pici - CP 6001
CEP 60455 - 760

Branco, Paulo
Instituto Superior Técnico –
Laboratório de Mecatrónica
Av. Rovisco Pais, 1096 Lisboa,
Portugal
pbranco@alfa.ist.utl.pt

Bravo, Diana
Dept. of Computer Science
Tijuana Institute of Technology
Tijuana, B. C., 22000
Mexico

Cândido, Marco
Pontifícia Universidade Católica do
Paraná
Programa de Pós-Graduação em
Informática Aplicada
Rua Imaculada Conceição, 1155,
CEP 80215-901, Curitiba, PR –
Brazil
candido@ppgia.pucpr.br

Cannavò, Flavio
Dipartimento di Ingegneria Elettrica,
Elettronica e dei Sistemi
Università di Catania
95125 Catania, Italy
fcannano@diees.unict.it

Castillo, Oscar
Dept. of Computer Science
Tijuana Institute of Technology
Tijuana, B. C., 22000
Mexico

Catucci, Graziano
Università degli Studi di Bari
Via Orabona 4, 70125 Bari, Italy
catucci@di.uniba.it

Cesario, Nicola
Automotive products Group - Power
Train Safety Division
Advanced System Applications R&D
STMicroelectronics, Italy
via R. De Feo,1, 80022, Arzano (NA)

Cha, Jianzhong
School of Mechanical and Electronical Control Engineering, Beijing
Jiaotong University, 100044 Beijing,
China

Chen, Kok Yeng
School of Electrical and Electronic
Engineering
University of Science Malaysia,
Engineering Campus
14300 Nibong Tebal, Penang,
Malaysia

Chen, Yen-Wei
Ritsumeikan University,
Shiga 525-8577, Japan
chen@is.ritsumei.ac.jp

Chiarella Gadaleta, Rita
Università degli Studi di Bari
Via Orabona 4, 70125 Bari, Italy
chiarella.rita@webmail.it

Chin Wei, Bong
Faculty of Computer Science and
Information Technology,
Universiti Malaysia Sarawak,
94300 Kota Samarahan, Sarawak,
Malaysia
cwbong@fit.unimas.my

Coelho, Leandro
Pontifícia Universidade Católica do
Paraná
Grupo Produtrônica, Pós-Graduação
em Engenharia de Produção e
Sistemas
Rua Imaculada Conceição, 1155,
CEP 80215-901, Curitiba, PR –
Brazil
leandro.coelho@pucpr.br

Covas, José
IPC - Institute for Polymers and Composites
Dept. of Polymer Engineering
University of Minho
Guimarães, Portugal

De Falco, Ivanoe
Institute of High Performance Computing and Networking – CNR
80131 Naples – Italy
ivanoe.defalco@na.icar.cnr.it

De Silva, A.K.M.
School of Engineering, Science and Design
Glasgow Caledonian University,
Glasgow, U.K.

Della Cioppa, Antonio
Department of Computer Science and Electrical Engineering
University of Salerno
84084 Fisciano (SA) – Italy
adellacioppa@unisa.it

Dong, Ming-Chui
Institute of System and Computer Engineering
Block 3, 1/F, University of Macau
1356, Taipa, Macau
dmc@inesc-macau.org.mo

Edan, Yael
Department of Industrial Engineering and Management
Ben-Gurion University of the Negev
84105, Beer-Sheva, Israel
yael@bgumail.bgu.ac.il

Embrechts, Mark J.
Department of Decision Sciences and Engineering Systems
Rensselaer Polytechnic Institute
Troy, NY 12180

Eryiğit, Gülşen
Istanbul Technical University
Computer Engineering Department
Istanbul, Turkiye
gulsen@cs.itu.edu.tr

Evangelista, Paul F.
Department of Systems Engineering
United States Military Academy
West Point, NY 10996

Fang, Peipei
Department of Computer Science
Georgia State University
Atlanta, GA 30302-3994 USA

Feng, Zhengjin
Institute of Mechatronics Control
Shanghai Jiao Tong University
Shanghai 200030, China

Ferreira, Cândida
Gepsoft, 73 Elmtree Drive
Bristol BS13 8NA
United Kingdom
candidaf@gepsoft.com

Fontanella, Francesco
Department of Information Engineering and Systems
University of Naples
80125 Naples – Italy
frfontan@unina.it

Górriz, Juan Manuel
Department Signal Theory and Communications
University of Granada, Spain
18071 Granada, Spain
gorriz@ugr.es

Gagliardi, Francesco
West–Sud S.r.l.
84091 Battipaglia (SA), Italy
francesco.gagliardi@libero.it

Gao, Junbin
School of Information Technology
Charles Sturt University
Bathurst, NSW 2795, Australia
jbgao@csu.edu.au

Gargaun, Florin
Department of Signal Circuits and Systems
University of Iasi
700050 Iasi, Romania
radu@altastiinta.ro

Gaspar-Cunha, António
IPC - Institute for Polymers and Composites
Dept. of Polymer Engineering
University of Minho
Guimarães, Portugal

Gil, C.
Dept. Arquitectura de Computadores y Electrónica
University of Almería, Spain
La Cañada de San Urbano s/n,
04120 Almería (Spain)
cgil@ace.ual.es

Gonçalves, Luís
IPC - Institute for Polymers and Composites
Dept. of Polymer Engineering
University of Minho
Guimarães, Portugal

Gonzalez, Claudia
Dept. of Computer Science
Tijuana Institute of Technology
Tijuana, B. C., 22000
Mexico

Gonzalez, Felma
Dept. of Computer Science
Tijuana Institute of Technology
Tijuana, B. C., 22000
Mexico

Groşan, Crina
Department of Computer Science
Faculty of Mathematics and Computer Science
Babeş-Bolyai University
Cluj-Napoca, 3400, Romania

Guaccero, Domenico
Università degli Studi di Bari
Via Orabona 4, 70125 Bari, Italy
guaccero@di.uniba.it

Hanqing, Zhao
Department of Automatic Control,
National University of Defense Technology,
Changsha 410073, Hunan, China
zhaohanqing@yahoo.com.cn

Harrison, D.K.
School of Engineering, Science and Design
Glasgow Caledonian University,
Glasgow, U.K.

Harrison, Robert F.
Dept of Automatic Control & Systems Engineering
The University of Sheffield
Mappin Street, Sheffield S1 3JD
United Kingdom

He, Li
Department of Computer Science and Engineering
Fudan University
200433 Shanghai, China
demonstrate@163.com

Hepburn, D.
School of Engineering, Science and Design
Glasgow Caledonian University,
Glasgow, U.K.

Herrera, Bruno
Cinq Technologies
Rua da Glória, 72, 8 andar, Centro Cívico
CEP 80030-060, Curitiba, PR – Brazil
bherrera@terra.com.br

Hien, Thai Duy
University of the Ryukyus,
Okinawa 903-0213, Japan
tdhien@augusta.eee.u-ryukyu.ac.jp

Howard, Ayanna
NASA Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109 USA
howard@robotics.jpl.nasa.gov

Huesca, Gabriel
Dept. of Computer Science
Tijuana Institute of Technology
Tijuana, B. C., 22000
Mexico

Iqbal, Asif
Pioneer Computers(P) Ltd.
Jamshedpur 831 001, India
asifiqbal_jsr@hotmail.com

Ji, Ye
Department of Computer,
Dalian University of Technology,
Dalian, 116023, China

Jiang, Fuhua
Department of Computer Science
Georgia State University
Atlanta 30302,USA
fjiang2@gsu.edu

Juhos, István
Department of Computer Algorithms and Artificial Intelligence
University of Szeged
6720 Szeged, Hungary
juhos@inf.u-szeged.hu

Kang, DongJin
Information Technology Services
Kyungpook National University
Daegu 702-701, Republic of Korea
djkang@mail.knu.ac.kr

Kartoun, Uri
Department of Industrial Engineering and Management
Ben-Gurion University of the Negev
Be'er-Sheva, Israel
kartoun@bgu.ac.il

Lee, Jay
Center for Intelligent Maintenance
University of Wisconsin at Milwaukee
Milwaukee
WI 53211, U.S.A

Lee, SangJo
Dept. of Computer Engineering
Kyungpook National University
Daegu 702-701, Republic of Korea
sjlee@mail.knu.ac.kr

Li, Bing Nan
Institute of System and Computer Engineering
Block 3, 1/F, University of Macau
1356, Taipa, Macau
bingoon@ieee.org

Li, Bo
Department of Computer,
Dalian University of Technology,
Dalian, 116023, China

Li, Zhiyi
Department of Computer Science
Georgia State University
Atlanta 30302,USA
zli4@student.gsu.edu

Lim, Chee Peng
School of Electrical and Electronic Engineering
University of Science Malaysia, Engineering Campus
14300 Nibong Tebal, Penang, Malaysia
cplim@eng.usm.my

Lim, SooYeon
Dept. of Computer Engineering
Kyungpook National University
Daegu 702-701, Republic of Korea
nadalsy@hotmail.com

Lima, Sanderson Emanuel U.
Electrical Engineering Department
Federal University of Ceara - Brazil
Campus Pici - CP 6001
CEP 60455 - 760

Liu, Chengliang
Institute of Mechatronics Control
Shanghai Jiao Tong University
Shanghai 200030, China

Liu, Hongbo
Department of Computer,
Dalian University of Technology,
Dalian, 116023, China
abrahamliu09@yahoo.com

Liu, Jian
Institute for Pattern Recognition and Artificial Intelligence
Huazhong Univ Sci and Tech,
Wuhan 430074, China
bliu@hust.edu.cn

Martin, Trevor P.
Department of Engineering Mathematics
University of Bristol
Bristol, BS8 1TR, UK
trevor.martin@bris.ac.uk

Martinez, Gabriela
Dept. of Computer Science
Tijuana Institute of Technology
Tijuana, B. C., 22000
Mexico

Martinoli, Alcherio
Swarm-Intelligent Systems Research Group
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
alcherio.martinoli@epfl.ch

Martins, Valdair
Pontifícia Universidade Católica do Paraná
Programa de Pós-Graduação em Informática Aplicada
Rua Imaculada Conceição, 1155,
CEP 80215-901, Curitiba, PR – Brazil
valdair@ppgia.pucpr.br

McGeough, J.A.
School of Engineering and Electronics
The University of Edinburgh,
Edinburgh, U.K.

Mediliyegedara, T.K.K.R.
School of Engineering, Science and Design,
Glasgow Caledonian University,
Glasgow, U.K.
kme2@gcal.ac.uk

Melin, Patricia
Dept. of Computer Science
Tijuana Institute of Technology
Tijuana, B. C., 22000
Mexico

Miao, Yubin
Institute of Mechatronics Control
Shanghai Jiao Tong University
Shanghai 200030, China

Montiel, Oscar
CITEDI-IPN
Tijuana, B. C., 22000
Mexico

Montoya, M.G.
Dept. Arquitectura de Computadores y Electrónica
University of Almería, Spain
La Cañada de San Urbano s/n,
04120 Almería (Spain)
mari@ace.ual.es

Mourelle, Luiza M.
Department of Systems Engineering and Computation
Faculty of Engineering, State University of Rio de Janeiro,
Rua São Francisco Xavier, 524,
Maracanã,
Rio de Janeiro, RJ, Brazil

Mukhopadhyay, Ananya
Research & Development Division
TATA STEEL
Jamshedpur 831 001, India
ananyamukhopadhyay@yahoo.co.in

Nakao, Zensho
University of the Ryukyus,
Okinawa 903-0213, Japan
nakao@augusta.eee.u-ryukyu.ac.jp

Nedjah, Nadia
Department of Electronics Engineering and Telecommunication
Faculty of Engineering, State University of Rio de Janeiro,
Rua São Francisco Xavier, 524,
Maracanã,
Rio de Janeiro, RJ, Brazil

Nunnari, Giuseppe
Dipartimento di Ingegneria Elettrica, Elettronica e dei Sistemi
Universita di Catania
95125 Catania, Italy
gnunnari@diees.unict.it

Oduguwa, Abiola
Dept. of Enterprise Integration
Cranfield University,
Bedfordshire, UK. MK43 0AL.
a.oduguwa@cranfield.ac.uk

Oliveira, José
Department of Production and Systems Engineering
University of Minho
4710-057 Braga, Portugal
zan@dps.uminho.pt

Ooi, Woi Seng
School of Electrical & Electronic Engineering
University of Science Malaysia
Engineering Campus, 14300, Nibong Tebal, Penang, Malaysia
ooiws@tm.net.my

Ortega, J.
Dept. Arquitectura y Tecnología de Computadores
University of Granada, Spain
C/ Periodista Daniel Saucedo s/n,
18071 Granada (Spain)
julio@atc.ugr.es

Palade, Vasile
Computing Laboratory,
University of Oxford,
Oxford, OX1 3QD, United Kingdom
vasile.palade@comlab.ox.ac.uk

Park, SeongBae
Dept. of Computer Engineering
Kyungpook National University
Daegu 702-701, Republic of Korea
seongbae@mail.knu.ac.kr

Petti, Palma
IIASS Istituto Internazionale per gli Alti Studi Scientifici
Vietri sul Mare, Salerno, Italy
pettip@libero.it

Pires, Armando
LabSEI, Escola Superior de Tecnologia de Setúbal
Instituto Politécnico de Setúbal
Rua Vale de Chaves, Estefanilha,
2914 Setúbal, Portugal
apires@est.ips.pt

Pirozzi, Francesco
Automotive Products Group - Power Train & Safety Division
Advanved System Applications R&D
STMicroelectronics, Arzano (NA)
80022 Italy

Puglisi, Giuseppe
Istituto Nazionale di Geofisica e Vulcanologia
Sezione di Catania
95125 Catania, Italy
puglisi-g@ct.ingv.it

Puntonet, Carlos Garcia
Departament Computer Architecture and Technology
University of Granada, Spain
18071 Granada, Spain
carlos@atc.ugr.es

Rafael, Silviano
LabSEI, Escola Superior de Tecnologia de Setúbal
Instituto Politécnico de Setúbal
Rua Vale de Chaves, Estefanilha,
2914 Setúbal, Portugal
srafael@est.ips.pt

Ranawana, Romesh
Computing Laboratory,
University of Oxford,
Oxford, OX1 3QD, United Kingdom
romesh.ranawana@comlab.ox.ac.uk

Rao, M.V.C.
Faculty of Engineering and Technology
Multimedia University
Melaka Campus, Jalan Ayer Keroh
Lama, Bukit Beruang, 75450 Melaka
Malaysia

Reis, Laurinda Lucia N.
Electrical Engineering Department
Federal University of Ceara - Brazil
Campus Pici - CP 6001
CEP 60455 - 760

Ribas, Leonardo
Cinq Technologies
Rua da Glória, 72, 8 andar, Centro Cívico
CEP 80030-060, Curitiba, PR – Brazil
lrribas@terra.com.br

Riyazuddin, Mohammed
Information and Technology Center
King Fahd University of Petroleum and Minerals
Dhahran, 31261, Saudi Arabia
mriyaz@kfupm.edu.sa

Roberto Sepulveda
CITEDI-IPN
Tijuana, B. C., 22000
Mexico

Roy, Rajkumar
Dept. of Enterprise Integration
Cranfield University,
Bedfordshire, UK. MK43 0AL
r.roy@cranfield.ac.uk

Rutkowski, Jerzy
Silesian University of Technology,
Institute of Electronics
ul. Akademicka 16,
44-100 Gliwice, Poland
jerzy.rutkowski@polsl.pl

Saad, Ashraf
School of Electrical and Computer Engineering
Georgia Institute of Technology
Savannah, GA 31407, USA
ashraf.saad@gtsav.gatech.edu

Sarfraz, Muhammad
Information and Computer Science Department
King Fahd University of Petroleum and Minerals
Dhahran, 31261, Saudi Arabia
sarfraz@kfupm.edu.sa

Saxena, Abhinav
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332, USA
asaxena@ece.gatech.edu

Semeraro, Giovanni
Università degli Studi di Bari
Via Orabona 4, 70125 Bari, Italy
semeraro@di.uniba.it

Shneor, Ran
Department of Industrial Engineering and Management
Ben-Gurion University of the Negev
84105, Beer-Sheva, Israel
ron2or@walla.com

Sierakowski, Cezar
Pontifícia Universidade Católica do Paraná
Grupo Produtrônica, Pós-Graduação em Engenharia de Produção e Sistemas
Rua Imaculada Conceição, 1155, CEP 80215-901, Curitiba, PR – Brazil
cezars@globo.com

Song, MuHee
Dept. of Computer Engineering
Kyungpook National University
Daegu 702-701, Republic of Korea
mhsong@mail.knu.ac.kr

Stern, Helman
Industrial Engineering and Management Department
Intelligent Systems Division
Ben-Gurion University of the Negev
84105 Be'er-Sheeva, Israel

Sun, Tong
Department of Computer,
Dalian University of Technology,
Dalian, 116023, China

Szarvas, György
Research Group on Artificial Intelligence
Hungarian Academy of Science, Szeged
6720 Szeged, Hungary
szarvas@inf.u-szeged.hu

Szymanski, Boleslaw K.
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180

Tan, Shing Chiang
Faculty of Information Science and Technology
Multimedia University
Melaka Campus, Jalan Ayer Keroh Lama, Bukit Beruang, 75450 Melaka
Malaysia
sctan@mmu.edu.my

Tantuğ, A. Cüneyd
Istanbul Technical University
Computer Engineering Department
Istanbul, Turkiye
cuneyd@cs.itu.edu.tr

Tao,Wu
Department of Automatic Control,
National University of Defense
Technology,
Changsha 410073, Hunan, China
wutao.nudt@hotmail.com

Tarantino, Ernesto
Institute of High Performance
Computing and Networking – CNR
80131 Naples – Italy
ernesto.tarantino@na.icar.cnr.it

Tay, Kai Meng
School of Electrical and Electronic
Engineering,
University of Science Malaysia
Engineering Campus, 14300 Nibong
Tebal, Penang, Malaysia
tkaimeng@yahoo.com

Tian, Jinwen
Institute for Pattern Recognition
and Artificial Intelligence
Huazhong Univ Sci and Tech,
Wuhan 430074, China
jwtian@hust.edu.cn

Tiwari, Ashutosh
Dept. of Enterprise Integration
Cranfield University,
Bedfordshire, UK. MK43 0AL.
a.tiwari@cranfield.ac.uk

Tunstel, Edward
NASA Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109 USA
tunstel@robotics.jpl.nasa.gov

Vai, Mang I
Faculty of Science and Technology
University of Macau
Taipa 1356, Macau
fstmiv@umac.mo

Valdez, Fevrier
Dept. of Computer Science
Tijuana Institute of Technology
Tijuana, B. C., 22000
Mexico

Vrânceanu, Radu
Department of Signal Circuits and
Systems
University of Iasi
700050 Iasi, Romania
radu@altastiinta.ro

Wachs, Juan P.
Industrial Engineering and Management Department
Intelligent Systems Division
Ben-Gurion University of the Negev
84105 Be'er-Sheeva, Israel

Weicker, Karsten
Department of Computer Science,
Mathematics and Natural Sciences
HTWK Leipzig University of Applied
Sciences
04251 Leipzig, Germany
weicker@imn.htwk-leipzig.de

Witold Pedrycz
Systems Research Institute
Polish Academy of Sciences, Warsaw,
Poland
pedrycz@ee.ualberta.ca

Xie, Kun
Institute of Mechatronics Control
Shanghai Jiao Tong University
Shanghai 200030, China

Yager, Ronald R.
Machine Intelligence Institute, Iona
College
New Rochelle, NY 10801
yager@panix.com

Zha, Xuan F.
National Institute of Standards and
Technology
Gaithersburg, MD 20899, U.S.A

Zhang, Junping
Shanghai Key Laboratory of
Intelligent Information Processing
Department of Computer Science
and Engineering
Fudan University
200433 Shanghai, China

Zhang, Lei
School of Mathematics, Statistics
and Computer Science
The University of New England
Armidale, NSW 2351, Australia
lzhang8@turing.une.edu.au

Zhang, Yan-Qing
Department of Computer Science
Georgia State University
Atlanta 30302,USA
zhang@cs.gsu.edu

Zhang, Yizhen
Engineering Design Research
Laboratory
California Institute of Technology
Pasadena, California 91125, USA
yizhen@caltech.edu

Zhao, Cheng
Department of Mathematics and
Computer Science, Indiana State
Univer-sity, Terre Haute, IN 47809,
U.S.A.

Zhao, Yong
School of Mechanical and Electronical Control Engineering, Beijing
Jiaotong University, 100044 Beijing,
China

Zheng, Sheng
China Three Gorges University
Yichang 443002, China
zsh@ctgu.edu.cn

Zhou, Zhi-Hua
National Laboratory for Novel
Software Technology
Nanjing University
210093 Nanjing, China
zhouzh@nju.edu.cn

Zielinski, Lukasz
Silesian University of Technology,
Institute of Electronics
ul. Akademicka 16,
44-100 Gliwice, Poland
lukasz.zielinski@comverse.com,

Part I

Plenary Presentations

Applying Fuzzy Sets to the SemanticWeb: The Problem of Retranslation

Ronald R. Yager

Machine Intelligence Institute, Iona College
New Rochelle, NY 10801
yager@panix.com

Abstract: We discuss the role of Zadeh's paradigmof computing with words on the semantic web We describe thethree important steps in using computing with words. We focuson the retranslation step, selecting a term from our prescribedvocabulary to express information represented using fuzzy sets. A number of criteria of concern in this retranslation processare introduced. Some of these criteria can be seen to correspondto a desire to accurately reflect the given information. Othercriteria may correspond to a desire, on the part provider ofthe information, to give a particular perception or "spin."We discuss some methods for combining these criteria to evaluatepotential retranslations.

Keywords: Computing with Words, Fuzzy Sets, Linguistic Approximation

1. Computing with Words and the SemanticWeb

The Semantic Web is invisioned as an extension of the currentweb in which information is given well-defined meaning and semantics,better enabling computers and people to work in cooperation. Among its goals is a humanlike automated manipulation of theknowledge contained on the web. While computers are good atprocessing information, they have no understanding of the meaningand semantics of the content which greatly hinders human likemanipulation. The fulfillment of the vision of the SemanticWeb requires tools that enable the computational representationof knowledge that emulates human deep understanding. Hence enablingintelligent information processing. Fuzzy subset theory andthe related paradigm of computing with words [1-3] providetools of this nature and hence will help to enable the automatedmanipulation of human knowledge.

The starting point for Zadeh's paradigm ofcomputing with words (CW) is a collection of human knowledgeexpressed in a natural language. The fundamental position takenby Zadeh is that the knowledge contained in a natural languageproposition can be viewed as an constraint on one or more ofthe implicit variables. The first step in the CW methodologyconsists of a translation (or explication) of these propositionsinto a formal computer manipulable language which he calls **G**eneralized **C**onstraint **L**anguage (GCL). The second step in the process is agoal directed manipulation of these propositions. This step,which Zadeh calls granular computing, can be seen as a kind ofinference process. This inference process is based on a constraintpropagation mechanism. The result of this second step is a propositionin GCL providing a constraint on a variable of interest. Thefinal step is a process of retranslation, here we convert a statementin GCL into an appropriate statement in natural language. Figure#1 provides a schematic view of the process of CW.

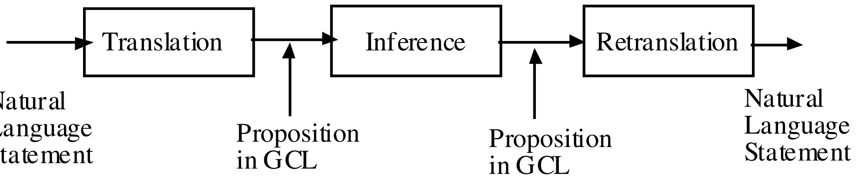


FIGURE #1. Computing with Words Paradigm

Step three, retranslation, involves the process of converting a statement in GCL into a proposition in natural language. It is aspects of this task that we shall investigate in this work. We describe and formalize the process of retranslation. We then discuss a number of objectives that may be involved in the process of selecting an appropriate retranslation. We first consider objectives related to the faithfulness of the chosen retranslation to the original proposition. We next look at some features, such as specificity and fuzziness, characterizing the nature of a potential retranslation. We next consider the issue of providing a retranslation that may support a particular point of view or perception we want to convey. As the retranslation process can be viewed as one of multicriteria decision making we next turn to discussing some methods for constructing multi-criteria decision functions.

2. The Retranslation Process

Let V be an attribute variable which has as its value some unique element in the set X, the universe of V. As our point of departure for the process of retranslation we shall assume that as a result of our inference process we obtain the proposition V *is* A where A is a fuzzy subset of the universe X. Our concern is to express this with a natural language statement.

Let L be a collection of terms from a natural language. We call L our natural language vocabulary. Formally the problem of retranslation is one of replacing the proposition V *is* A by VÊ *is* L where L is some element from the natural language vocabulary L . As an aid in this process we have at our disposal a collection F of fuzzy subsets of X such that for each fuzzy subset F in F the proposition V *is* F can be seen as a faithful translation of the knowledge associated with a statement of the form V *is* L where L an element for L. Each fuzzy subset in F corresponds to a unique natural language term from L. We assume that each F is normal, that is there exists an x such that F(x)Ê = 1. We let F_L indicate the subset of F that corresponds to the linguistic term L.

The process of retranslation is one of substituting the proposition V *is* F for V *is* A where F is some element from F and then expressing the output as V *is* L where L is the linguistic term associated with F. The key issue in this process is the substitution of V *is* F for V *is* ÊA.

The process of replacing A with F can be seen to depend on two classes of criteria. The first class depending the "faithfulness" of F to A. This basically involves using some criteria measuring aspects of the relationship of A and F. The second class concerns itself with the possession of some special features by the replacement set. The criteria involved here may not involve the relationship between F and A but may involve the relationship of F to some external condition.

3. Determining the Validity of a Retranslation

The most important requirement involved in the retranslation process is validity or truth. We don't want the process of retranslation to result in a statement that is not true. For example, in the case of John's age, if our inference process results in a crisp subset A = {x|ÊxÊ = 1 to 20}, we don't want to retranslate this into John is a *senior*.

The proposition V *is* A is essentially an indication that the value of V lies in the subset A. It specifies the possible values for V. Consider, for example, the situation where V is John's age and AÊ = {10, 11, 12, 13, 14}. Here we are informed that the possible ages for John lie in this set. To translate this into "John is less than 20" is valid. On the other hand, to say that John is over 50 is clearly an invalid retranslation of our information. To say that John is 13, while not incorrect, is not justified by our knowledge and hence not valid.

The central issue here involves what types of statements we can truthfully infer from our given knowledge, V *is* A. The entailment principle introduced by Zadeh [4] provides an important tool for addressing this issue. Zadeh's

entailment principle states that from the proposition VÊ*is* A we can truthfully infer V *is* F where A ⊆ F. Thus in associating V *is* A with only those statements V *is* F satisfying this condition assures of making true statements (valid statements) based on our knowledge.

In [5] Zadeh suggested a definition for containment. Assume A and F are two fuzzy subsets of X we say that A ⊆ F if A(x) ≤ F(x) for all x ∈ X. Here we point out that Zadeh's definition of containment is crisp, either A is contained in F or A is not contained in F. Using this definition for containment gives a very crisp interpretation of the entailment principle. A softening of the entailment principle can be had by softening the definition of containment. With a softer definition of containment we can get a degree to which AÊ ⊆ ÊF. Using this softer definition in our entailment principle we can get a degree to which the statement V *is* A entails the statement V *is* F. This can allow us to obtain for each proposition VÊ *is* F a degree to which it satisfies the requirement of being a valid inference for V *is* A.

A number of "softer" definition of inclusion have been suggested in the literature. Here we note that pioneering work of Bandler and Kohout [6]. In the following we shall let Deg(A ⊆ F) indicate the degree to which A is a subset of F. Where Deg(A ⊆ F) ∈ [0, 1]. We note that Zadeh's definition has Deg(A ⊆ F) ∈ {0, 1}.

Bandler and Kohout [6] provided a general definition for thedegree of inclusion as

$$\mathrm{Deg}(A \subseteq F) = \underset{x \in X}{\mathrm{Min}}[I(A(x), F(x))]$$

where the operator I is an implication operator. Formally wedefine operator I as a mapping I:Ê[0,Ê1]Ê× [0, 1] → [0, 1] having the following properties:

1. $I(a_1, b) \geq I(a_2, b)$ if $a_2 \geq a_1$
2. $I(a, b_1) \geq I(a, b_2)$ if $b_1 \geq b_2$
3. $I(0, b) = 1$
4. $I(a, b) = 1$ if $b \geq a$

We note that I is not symmetric, I(a, b) ≠ I(b, a)

Some forms of this operator are the following:

Gaines-Rescher: $I(a, b) = \begin{cases} 1 & a \leq b \\ 0 & a > b \end{cases}$

Godel: $I(a, b) = \begin{cases} 1 & a \leq b \\ b & a > b \end{cases}$

Goguen: $I(a, b) = \mathrm{Min}(1, \frac{b}{a})$

Lukasiewicz: I(a, b) =Min(1, 1 - (a - b)) = Min(1, 1 + (b - a))

We note the Gaines-Rescher leadsto Zadeh's definition.

Whatever approach we take the fundamental idea is that we can associate with each F ∈ F a degree to which it is valid to infer V *is* F from V *is* A. Since our final objective is to select some linguistic term L from L we can obtain the validity of the term L given V is A, Val(L/A) as $\mathrm{Max}_{F \in \mathsf{F}_L}$(Deg (A ⊆

F)). This follows since for any F ∈ F_L we express this as the natural language statement V *is* L.

4. Measuring the Closeness of Fuzzy Subsets

Another possible consideration in determining the appropriateness of replacing V *is* A with V *is* F is whether the fuzzy subset F is close to the fuzzy subset A. In order to capture this we need some relationship *Close* such that for any two fuzzy subsets A and F, *Close(*A, F) ∈ [0, 1] indicates the degree to which A and F are close.

While closeness may not necessarily be a required condition in determining the retranslation but in some cases it may be desired by the user.

The closeness of F to A is related to the distance between these two sets. Early work on the distance between fuzzy subsets was presented in Kaufmann [7]. The definition of the distance between two fuzzy subsets depends on whether the universe is finite ornot. Here we shall assume X is a finite universe, X = $\{x_1, x_2, ..., x_n\}$. The fundamental unit in discussing closeness is the elementary difference $\Delta_j = |A(x_j) - F(x_j)|$. We note $\Delta_j \in [0, 1]$. Using this we can obtain the Hamming distance $D_1(A, F) = \sum_{j=1}^{n} \Delta_j$. From this we can obtain *Close*(A, F) = $1 - \frac{1}{n}\sum_{j=1}^{n} \Delta_j$. It is the complement of the average of absolute differences in membership. We note *Close*(A, F) ∈ [0, 1].

A more general measure of closeness can be obtained if we use the generalized mean

$$Close(A, F) = 1 - \left(\frac{1}{n}\sum_{j=1}^{n} \Delta_j^{\alpha}\right)^{1/\alpha}$$

where $\alpha \geq 1$. We note for $\alpha = 1$ we get the normalized Hamming. For $\alpha = 2$ we get

$$\text{Close}(A, F) = 1 - \frac{1}{\sqrt{n}}\left(\sum_{j=1}^{n} \Delta_j^2\right)^{1/2}$$

and for $\alpha \to \infty$we get *Close(*A, F) - 1-$\text{Max}_j[\Delta_j]$.

In the preceding we have essentially introduced a definition for a relationship Distant, where

$$\text{Distant}(A, F) = \left(\frac{1}{n}\sum \Delta_j^{\alpha}\right)^{1/\alpha}$$

and then expressed *Close* as not Distant: *Close*(A, F) = 1 - Distant(A, F)

5. Measuring the Fuzziness and Specificity

In addition to desiring that the retranslation V *is* F is valid and close to our knowledgethat V *is* A we may alsodesire that V *is* F havesome other features. One such feature is related to the crispnessor fuzziness of the retranslated value. The concept of fuzzinessis essentially related to the type of boundary distinguishingmembership from non-membership in F. For example to say thatJohn is between 22 and 28 is very crisp while saying that heis about 25 is fuzzy.

De Luca and Termini first provided a measureof fuzziness [8]. Their definition was greatly influenced byShannon's concept of entropy. A measure of fuzziness of a subsetF, FUZZ(F) $\in [0, 1]$, shouldsatisfy the following conditions:

P_1: FUZZ(F) attainsits minimum value, FUZZ(F) = 0, if F is a crisp set, F(x) $\in \{0, 1\}$

P_2: FUZZ(F) attainsits maximum value for the fuzzy subset F(x) = 1/2 for all $x \in X$

P_3: FUZZ(F) $\geq$ FUZZ(F*) if F* is a sharpenedversion of F, that is if F*(x) $\geq$ F(x) when F(x)Ê$\geq$ 1/2 and F*(x) $\leq$ F(x) if F(x) $\leq$ 1/2

At an intuitive level the concept of fuzziness is related tothe lack of distinction between a set and its negation. Usingthis Yager[9] suggested a measure of fuzziness based on the lackof distinction between F and $\bar{F}$, the closer F and $\bar{F}$ the more fuzzy.

We can measure the distance between a fuzzy set F and its negation$\bar{F}$ using the Hamming

distance, $D(F, \bar{F}) = \sum_{j=1}^{n} \left|F(x_j) - \bar{F}(x_j)\right| = \sum_{j=1}^{n} \left|2F(x_j) - 1\right|$. Normalizing this and taking it complement gives us

$$\mathrm{FUZZ}(F) = 1 - \frac{1}{n}\left(\sum_{j=1}^{n} \left|2F(x_j) - 1\right|\right)$$

We can easily show this definition satisfies the threeconditions of De Luca and Termini and lies in the unit interval. In particular if $F(x_j) = 1/2$ for all x_j then FUZZ(F) = 1. If $F(x_j) = 1$ or 0 then $\left|2F(x_j) - 1\right| = 1$ and hence if $F(x_j) = 1$ or 0 for all x_j then FUZZ(F) = 0.

More generally we can use any measure of distance between fuzzysubsets to define a measure of fuzziness. Thus if Dist(A, B)is a measure of distance between two fuzzy subsets then we candefine a measure of fuzziness as FUZZ(F) = 1 - Dis(F, $\bar{F}$)

It is important to emphasize that this criteria just depends upon F, it doesn't involve the relationship between F and A. It is just a condition about F.

Another feature associated with a granular object that is of concern in the process of linguistic retranslation is the concept of **specificity** of a fuzzy subset. The concept of specificity [10] is related to the quantity of information or certainty associated with a linguistic value. For example saying that Rachel is 32 is more informative than saying that she is in her early thirties. Essentially the specificity points to the degree to which a fuzzy subset contain one and only one element. We note that whereas fuzziness is related the type of boundary, the concept of specificity is related to more closely related to the cardinality or width of the set.

Definition: Assume F is a fuzzy subset of the universe X. A measure of the specificity of F, denoted Sp(F), is a value lying in the unit interval that satisfies the following properties.

1) Sp(F) = 1 if there exists one and only one $x^* \in$ F with F(x*) = 1 and F(x) = 0 for all xÊ-Êx*

2) If F(x) = 0 for all x then Sp(F) = 0

3) F_1 and F_2 are normal fuzzy subsets and $F_1(x) \geq F_2(x)$ for all x then $Sp(A_1) \leq Sp(A_2)$

4) If F_1 and F_2 are crisp subsets such that Cardinality(A_1) ≤ Cardinality(A_2) then Sp(A_1)Ê ≥ ÊSp(A_2).

In the framework of normal sets specificity can be seen as related to the size or cardinality of the set. We must carefully distinguish between specificity and fuzziness. A set can be crisp and non-specific, for example saying that John's over 10 years old is crisp but not very specific. Thus specificity is related to width of the set. On the other hand "close to 20" is specific but fuzzy.

While a number of measures of specificity has been suggested in the literature a particularly useful is the following. Let $F(x^*) = \mathrm{Max}_X F(x)$, x* is an element in X where F attains its largest membership grade. Using this we measure the specificity of F as

$$Sp(F) = F(x^*) - \frac{1}{n-1} \sum_{\substack{j=1 \\ j \neq x^x}}^{n} F(x_j)$$

This measures the specificity as the difference between the largest membership grade and the average of the remaining membership grades.

Here again we point out that specificity is a characteristic solely of F, and is not dependent on the knowledge V *is* A. It is just a function of the proposed target value.

6. Providing Retranslations that Give Particular Perceptions

In some cases a user may have some goal or agenda in the retranslation process. The user of such a system may want to put a certain" spin" on the output of the system, they may want to convey a certain impression of the state of the world. For example, a cigarette company in expressing information about heart disease and cigarette smoking clearly to reflect a certain point of view. A company trying to sell a financial product will try to express the information about the risk involved in the most favorable light.

Thus, in addition to requiring that the retranslation satisfy some property such as being truthful, a user may want to include in the retranslation process a criteria reflecting a particular point of view. In order to include such criteria we must investigate how to measure the degree to which a particular retranslation conveys an impression or perception about the state of the world that the user desires to convey.

Let V be a variable with universe X. Let P be a fuzzy subset on the universe X corresponding to some preference of a **P**rovider **o**f **I**nformation(POI) with regard to expressing the knowledge about V. The **POI**, the user of the retranslation process, desires to convey the perception that the state of the world is V *is* P. For example if V corresponds to the risk associated with a financial instrument, such as a bond, the POI may want to convey the perception that V *is* LOW.

Let F be some fuzzy subset corresponding to some natural language term from our vocabulary. We are interested here in formulating a criteria to measure the degree to which the use of statement V *is* F conveys the impression that the state of the world is V *is* P. Before proceeding we must keep in mind that usually in the retranslation process we are required to satisfy many criteria among which the conveying of an impression is only one. One criteria that is almost always present is the need of truthfulness. The user of these systems, especially if they are responsible organizations, are usually bound to provide information that is truthful.

Let us now look at the issue of conveying the impression V *is* P by dispensing the information V *is* F. If F = P then the statement V *is* F definitely conveys the perception that VÊ *is* ÊP. Another method of conveying the desired perception is for the statement V *is* F to be such that one can logically infer V *is* P. For example, if we are interested in conveying the impression it is "cold" outside saying that the temperature is zero degrees certainly conveys the impression.

As we have previously noted the entailment principle tells us that the statement that V *is* F allows us to infer with certainty that V *is* P is true if $F \subseteq P$. Thus one way to convey with certainty the perception that V *is* P is to provide the information V *is* F where $F \subseteq P$.

Sometimes because of our restriction that the provided by the systems, V *is* F, is true with respect to the actual state of the world, V *is* A, we can't use a statement V *is* F that allows us with certainty to convey the perception V *is* P, that is we can't use F $\subseteq$ P. For example, assume V indicates the degree of risk associated with a financial instrument measured on a scale of zero to one. Here the lower the value the less the risk. Let the situation by such that A is *about 0.5*. Here we can't truthfully make a statement that allows us with certainty to infer that V is low, that is we can't make a statement V *is* F*where F* $\subseteq$ LOW and still retain the fact that V *is* "about 0.5." That is we can't have F* $\subseteq$ Low and {0.5} $\in$ F*.

In these situations we must lower our ambitions. Since we can't make a truthful statement that would allow the logical conclusion that V *is* P, we must look to make a statement V *is* F that makes it as easy as possible for somebody to **draw the conclusion** that V *is* P is true. The term drawing of a conclusion as we are using it is meant indicate a process in which one makes some inference from a fact or observation that is not necessarily logically certain.

In order to facilitate a receiver of information to draw the conclusion we desire, we must take advantage of the human cognitive component in the reasoning process. Specifically human reasoning and inference is not purely deductive and objective but it has a strong subjective component which uses induction and abduction among other mechanisms. It is also strongly influenced by the preferences, expectations and desires of the reasoner. Our purpose here is not to get into a discussion of human reasoning but just to note that human reasoning is a complex process that is often subjective and even goal directed.

Here we shall take a simplistic look at the human reasoning process with an eye toward understanding how we can effect and enable the process of having a receiver of the information VÊ *is* F draw the conclusion that VÊ *is* ÊP. We shall start with a given proposition V *is* F and define a number processes associated with making inferences from this assumed state of the world. First we recall that the statement V *is* F essentially indicates thatthe value of the variable V lies in the set F. More formally V *is* F induces a possibility distribution such that F(x) is the possibility that V = x.

If E be a subset of X such that F $\subseteq$ E the process of infering that V *is* E from V *is* F is called **deduction**. As we have already indicated deduction is a valid and sound logical operation. Here we are saying that since the value of V is contained in the set F it is also surely contained in the larger set E (see Figure #2)

Consider now the situation in which we have another subset H such that H $\cap$ F = $\varnothing$, they have no elements in common (see Figure #3).

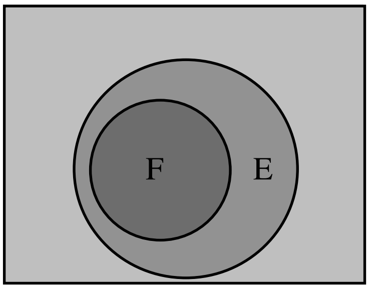


Figure #2. Deduction: Concluding E from F

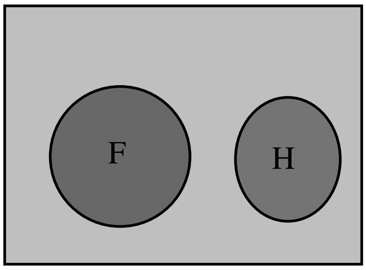


Figure #3. Contradiction: Concluding H from F

It is clear that in this situation drawing the conclusion that V *is* H from the knowledge that V *is* F is wrong, it can never give you the correct answer. We shall call this process **contradiction**. What is clear that being "rational" implies we never make a contradiction

Let us now consider the situation where B is a subset such that B ⊂ F (see figure #4).

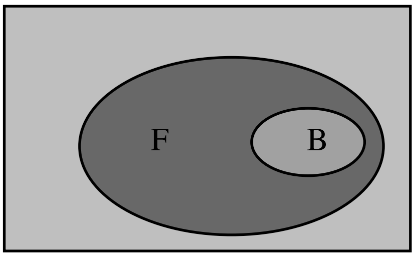


Figure #4 Reduction: Concluding B from F

In this case we shall use the term **reduction** to indicate the process of drawing the conclusion VÊ *is* ÊB from V *is* F. We note that while the process of reduction is not a sound logical process its an often used in human reasoning[1].

[1] A well know exampleof this occurs when informed that an animal is a bird one assumesit flies. Being informed that somebody lives in New York Cityoften leads people to conclude they live in Manhattan

The process of reduction plays a central role in non-monotonic reasoning [11]. The process of reduction provides a pragmatic aid to enable human beings to act in the face of uncertainty, it is an uncertainty reducing method. Decision making of many types involves aspects of reduction. The point here is that reduction is an often used pragmatic mode of human reasoning.

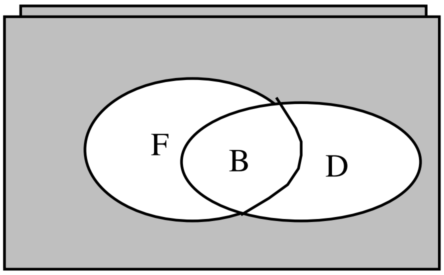
As we shall subsequently see, reduction is the reasoning mechanism that plays a central role in the process of trying to get people to draw the conclusion that V *is* P from V *is* F. We shall see that the larger the portion of F that P constitutes the easier it is to convey the impression that V *is* P given V *is* F.

At a formal level we see that reduction is based on a valid reasoning process where in the context of figure #4 starting with V *is* F we can correctly conclude V *is* B is possible given V *is* F [12]. Then reduction essentially involves goes beyond this by changing what is only possible to making it certain. That is, by introducing artificial certainty we change V *is* B is possible to obtain V *is* B.

We observe that using reduction, while not purely logical, can be seen as pragmatic and not irrational. It makes dealing with the world simpler. It has many extra logical uses among which is anxiety reduction, it simplifies the world. Our point here is not to justify this operation but only to acknowledge its existence.

While reduction and deduction are the mechanisms that will playa central role in our understanding the process of conveying a particular impression we shall take the opportunity of having introduced these ideas to consider some related formal issues.

Closely related to reduction is what we shall call **transduction** (see figure #5). LetD be a subset such that F $\cap$ D = B $\neq \varnothing$ and F $\not\subset$ D. The process of drawing the conclusion that V *is* D from V *is* F we shall call **transduction** (see figure #5).



Figure# 5. Transduction: Concluding D from F

Here then D has some elements in common with F, B. We note that transduction can be viewed as a combination of reduction and deduction. In transduction we first us reduction to draw the conclusion V *is* B from V *is* F and then use deduction to extend V *is* B to V *is* D. We shall prefer to keep this as one operator. We note that a distinction between transduction and a closely related application of deduction. If we concluded from V *is* F that V *is* F $\cup$ D then we would be just simply using deduction. Here with transduction we have removed F - B from our conclusion.

Here we point out some concerns and problems that can arise using multiple steps of reduction or the related operation of transduction. As the following illustrates multiple users of these "pragmatic" non-contradictory processes can lead to a contradiction. Consider the following. Assume we start with V *is* F. Assume B ⊂ F and let D = B ∪ E where E ∩ F = ∅. Then using transduction, inferring V *is* B by reduction of V *is* F and then using deduction from V *is* B we can conclude V *is* D. Furthermore again using reduction on V *is* D we can conclude V is E. However since E ∩ F = ∅ we have inferred something clearly contradictory to our knowledge that V *is* F.

While most people would not directly infer V *is* E from V *is* F, the step by step nature of the above process can mask the irrationality of the result. Thus reduction can lead to difficulty. Nevertheless reduction is part of the human reasoning machinery. Its use reflects a trade off between its pragmatic uncertainty and complexity reducing function and its potential for leading to invalid conclusions.

The point we want to make here is the following. A purely logical reasoner will only use deduction. However human beings because for pragmatic reasons have found it beneficial to use the non-logical operation of reduction. As we shall see it is this type of reasoning that provides a mechanism to enable us to induce people to draw a conclusion satisfying a certain perception we desire.

We recall reduction is essentially a process were we conclude some subset from a set containing it (see figure #4). Let us try to formalize a measure that can be used to quantify the ease with which one can draw the conclusion that V *is* P given the statement VÊ *is* F. We shall call this the **Impression of P given F**, and denote it as Imp(P|F). If we can infer from V *is* F some statement V *is* G where G ⊆ P then we draw the conclusion we want. Essentially we are faced with the problem of eliminating that part of F that is not in P. Thus the "difficulty" associated with the desired reduction is related to how difficult it is to eliminate the part of F that is not in P. For our purposes a simple useful measure of ease (or naturalness) of the reduction process is

$$\mathrm{Imp}(P/F) = \frac{|F| - |\overline{P} \cap F|}{|F|} = 1 - \frac{|\overline{P} \cap F|}{|F|}$$

Let us look at this measure. For simplicity, we shall assume all sets are crisp. Consider the case where F ⊆ P. In this case $\overline{P} \cap F$ = ∅ and Imp(P|F) = 1. This is as it should be for this corresponds to the situation of deduction. At the other extreme, consider the case where F ∩ P = ∅, there are no elements in common between F and P. In this case where F ∩ P = ∅ then all elements in F are in $\bar{P}$ hence $\bar{P} \cap F = F$ and we get Imp(P/F) = $1 - \frac{|F|}{|F|} = 0$. This is again as it should be, the ease of concluding V *is* P from V *is* F is zero. For we recall that with $F \cap P = \varnothing$ inferring P from F would be a clear contradiction. Finally in the case when $P \cap F \neq \varnothing$ and $F \not\subset P$ we get for Imp(P/F) the proportion of elements in F that are also in P. Alternatively $\left|\frac{\bar{P} \cap F}{F}\right|$ is the number of elements we must eliminate from F in order to obtain a subset contained in P.

What we are advocating here is that Imp(P/F) can provide a measure which can be used to quantify the ease with which one can perform this reduction. Thus Imp(P/F) provides a measure to compute the degree to which a subset F lets us draw the conclusion P. It allows us to evaluate the criteria "our retranslation should make it easy for the receiver of the information to draw the perception P."

One often used technique that serves this purpose of conveying P is to imbed our knowledge of what is known to be true, V *is* A, in a broader statement, V *is* F, that easily gives the impression that V *is* P. Let us see if our model correlates with our observation of what is often done. Again for simplicity we restrict ourselves to the crisp environment. Assume we have a information system that infers VÊ *is* A. Our problem is to select some proposition V *is* F from our vocabulary L to make it as easy as possible for a recipient of the information V *is* F to draw the conclusion that V *is* P. The one constraint here is that the system can't lie, its output must satisfy $A \subseteq F$. Thus the expressed proposition F must be in L and be of the form $F = A \cup B$ where $A \cap B = \varnothing$. The subset B can be decomposed into two components, $B = B_1 \cup B_2$, those not contained in P, B_1, and then those contained in P, B_2 . Using this we get $F = A \cup B_1 \cup B_2$.

7. Multicriteria Evaluation

Assume we have some process of soft computing which leads to an output, V *is* A, where A is a fuzzy subset of X. Let F be the collection of fuzzy subsets corresponding to the linguistic terms available to us. As part of the retranslation process we must pick some fuzzy subset from this collection. In the preceding sections we described a number of criteria and features that may taken into consideration in this process. Let $C_1, ..., C_q$ be a collection of criteria that are of interest to the agent responsible for making the linguistic retranslation of V *is* A. Here using the procedures described earlier we assume that for any F ∈ F we are able to obtain $C_j(F)$, the degree to which F satisfies C_j. We note that in some cases $C_j(F)$ depends on A as well as F. This occurs for example when we want truthfulness in our retranslation. We can make this explicit by using the notation $C_j(F/A)$.

In this situation the selection of F becomes a multiple criteria decision problem. For each FÊ∈ÊF we calculate the overall satisfaction to the criteria, $S(F) = Agg(C_1(F), C_2(F), ..., C_q(F))$. We then select the F with largest satisfaction.

The formulation for Agg is a reflection of the desired relationship between the multiple criteria. A considerable body of literature exists in the fuzzy set literature on the formulation of these multiple criteria decision functions. Here we shall briefly describe some procedures for aggregated these criteria.

One technology that can be used to formulate the Agg function is based on the OWA operators [13]. Using the OWA operator we obtain

$$S(F) = Agg(C_1(F), ..., C_q(F) = \sum_{j=1}^{q} w_j b_j$$

Here the w_j are a collection of weights such that $w_j \in [0, 1]$ and $\sum_{j=1}^{q} w_j = 1$ and b_j is the j^{th} largest of the $C_i(F)$. We refer to the weights collectively by a vector W, called the OWA weightingvector, where $w_j$is the j^{th} component in W. We let B be a vector whose components are the b_j. We call B the ordered argument vector. Using these vectors $S(F) = W^T B$. The choice of W determines the type of aggregation being used. For example, if W is such that $w_q = 1$ and w_jÊ= 0 for all $j \neq q$ then S(F)Ê=Ê$b_q = Min_i(C_i(F))$. This is seen as modeling a situation in which the user desires ***all*** the specified criteria be satisfied when selecting the retranslation. If W is such that $w_1 = 1$ and w_j =0 for all $j \neq 1$ then S(F)= $Max_j(C_i(K)$. This models a user who desires ***any*** of the criteria be satisfied. Another special case is when $w_j = 1/q$ here $S(F) = \frac{1}{q}\sum_{i=1}^{q} C_i(F)$. By selecting different manifestations of W we can model different relationships between the criteria.

In the spirit Zadeh's idea of computing with words, Yager[14] suggested an approach to obtaining the weights using a linguistic specification of the aggregation imperative.. Let f:[0,Ê1]Ê→ [0, 1] be a function such that f(0) = 0, f(1) = 1 and f(x) ≥ f(y) if x > y. We call these **B**asic **U**nit interval **M**onotonic (BUM) functions. We can show that if we define

$$w_j = f(\frac{j}{q}) - f(\frac{j - 1}{q})$$

then for j = 1 to n we have $w_j \in [0,1]$ and $\Sigma_j\ w_j = 1$. Thus using this we can obtain a valid collection of OWA weights.

Using Zadeh's idea of computing with words and particularly the idea of linguistic quantifiers we can provide for the linguistically guided formulation of aggregations. Let Q be a linguistic quantifier such as "most," all" and "some." As suggested by Zadeh [15] we can represent these linguistic quantifiers as fuzzy sets Q of unit intervals. Here for any $x \in [0, 1]$,Q(x) indicates the degree to which the proportion x satisfies the concept of Q. An important class of these quantifiers are those which have the properties that: Q(0) = 0, Q(1) = 1 and are increasing Q(x)Ê≥ÊQ(y) for x > y. Examples of these are "most", "all", "any", "at least α %." Many of the quantifiers used to express aggregation imperatives are of this monotonic type, the more conditions satisfied the better.. These quantifiers are such that Q is a BUM function. We are now describe how we use these ideas to formulate aggregation functions.

Let $C_1, ..., C_q$ be a collection of criteria associated with the selection of the translation. Let Q be a linguistic qualifier, such as "most", indicating what portion of the criteria need to be satisfied, our aggregation imperative is noted as $Q(C_1, ..., C_q)$. As suggested in [14] we can implement this aggregation as follows. Let Q be the fuzzy subset of the unit interval corresponding to the linguistic quantifier Q. We use Q to calculate a weighting vector W such that $w_j = f(\frac{j}{q}) - f(\frac{j - 1}{q})$. We calculate the satisfaction of F to the criteria using an OWA aggregation S(F) = $OWA_W(C_1(F), ...,C_q(F))$.

8. Conclusion

We have focused on the retranslation stage in the paradigm of computing with words and suggested a framework that can be used to implement this process. In this framework we have formulated this as a multi-criteria decision problem. We introduced a number of criteria that can be used in the process of selecting there translation, clearly additional criteria may introduced in particular cases. It is worth noting that the criteria we have considered fall into three categories. The first category are those, such as validity and closeness, which are based on measuring some type of relationship between the given data and a proposed retranslation. The second category involves the measurement of some feature solely about the proposed retranslation, the measures of fuzziness and specificity are examples of this class. Finally the third requires the introduction of some additional concept and involves determining the relationship between a proposed retranslation and this exogenous concept. This third category of criteria is of particular importance in many types of information warfare. Here we maybe interested in providing disinformation(information that may lead an adversary to draw an inappropriate conclusion) while at the same time not placing ourselves in a position of being able to labeled as lying.

9. References

[1]. Zadeh, L. A., "Fuzzy logic = computing with words," IEEE Transactions on Fuzzy Systems 4, 103-111, 1996.

[2]. [2]. Zadeh, L. A., "From computing with numbers to computing with words- From manipulation of measurements to manipulations of perceptions," IEEE Transactions on Circuits and Systems 45,105-119, 1999.

[3]. Zadeh, L. A. and Kacprzyk, J., Computing with Words in Information/IntelligentSystems 1, Physica-Verlag: Heidelberg, 1999.

[4]. Zadeh, L. A., "A theory of approximate reasoning," in Machine Intelligence, Vol. 9, edited by Hayes, J., Michie, D. and Mikulich,L. I., Halstead Press: New York, 149-194, 1979.

[5]. Zadeh, L. A., "Fuzzy sets," Information and Control 8,338-353, 1965.

[6]. Bandler, W. and Kohout, L., "Fuzzy power sets and fuzzy implication operators," Fuzzy Sets and Systems 4, 13-30, 1980.

[7]. Kaufmann, A., Introduction to the Theory of Fuzzy Subsets: Volume I, Academic Press: New York, 1975.

[8]. [8]. DeLuca, A. and Termini, S., "A definition of a non-probabilistic entropy in the setting of fuzzy sets," Information and Control20, 301-312, 1972.

[9]. Yager, R. R., "On the measure of fuzziness and negation part I: membership in the unit interval," Int. J. of General Systems 5, 221-229, 1979.

[10]. Yager, R. R., "On measures of specificity," in Computational Intelligence: Soft Computing and Fuzzy-Neuro Integration with Applications, edited by Kaynak, O., Zadeh, L. A., Turksen, B.and Rudas, I. J., Springer-Verlag: Berlin, 94-113, 1998.

[11]. Ginsberg, M. L., Readings in Non-monotonic Reasoning, Morgan Kaufmann: Los Altos, CA, 1987.

[12]. Yager, R. R., "A generalized view of non-monotonic knowledge: A set theoretic perspective," Int. J. of General Systems 14,251-265, 1988.

[13]. Yager, R. R., "On ordered weighted averaging aggregation operators in multi-criteria decision making," IEEE Transaction son Systems, Man and Cybernetics 18, 183-190, 1988.

[14]. Yager, R. R., "Quantifier guided aggregation using OWA operators," International Journal of Intelligent Systems 11,49-73, 1996.

[15]. Zadeh, L. A., "A computational approach to fuzzy quantifiers in natural languages," Computing and Mathematics with Applications9, 149-184, 1983.

Granular Computing: An Overview

Witold Pedrycz
Systems Research Institute
Polish Academy of Sciences, Warsaw, Poland

Abstract In this study, we present a general overview of Granular Computing being regarded as a coherent conceptual and algorithmic platform supporting the design of intelligent systems. Fundamental formalisms of Granular Computing are identified and discussed. The linkages between Granular Computing and Computational Intelligence are revealed as well.

1. From Information Granules to Granular Computing

Information granules permeate human endeavors [1][14][20]-[23]. No matter what problem is taken into consideration, we usually cast it into a certain conceptual framework of basic entities, which we regard to be of relevance to the problem formulation. This becomes a framework in which we formulate generic concepts adhering to some level of abstraction, carry out processing, and communicate the results to the external environment. Consider, for instance, image processing. In spite of the continuous progress in the area, a human being assumes a dominant position when it comes to understanding and interpreting images. Surely, we do not focus our attention on individual pixels and process them as such but group them together into semantically meaningful constructs – familiar objects we deal with in everyday life. Such objects involve regions that consist of pixels or categories of pixels drawn together because of their proximity in the image, similar texture, color, etc. This remarkable and unchallenged ability of humans dwells on our effortless ability to construct information granules and manipulate them. As another example, consider a collection of time series. From our perspective we can describe them in a semi-qualitative manner by pointing at specific regions of such signals. Specialists can effortlessly interpret ECG signals. They distinguish some segments of such signals and interpret their combinations. Experts can interpret temporal readings of sensors and assess the status of the monitored system. Again, in all these situations, the individual samples of the signals are not the focal point of the analysis and the ensuing signal interpretation. We always granulate all phenomena (no matter if they are originally discrete or analog in their nature). Time is another important variable that is subjected to granulation. We use seconds, minutes, days, months, and years. Depending which specific problem we have in mind and who the user is, the size of information granules (time intervals) could vary quite dramatically. To the high level management time intervals of quarters of year or a few years could be meaningful temporal information granules on basis of which one develops any predictive model. For the designer of high-speed integrated circuits and digital systems, the temporal information granules concern nanoseconds, microseconds, and perhaps microseconds. Even such commonly encountered and simple examples are convincing enough to lead us to ascertain that (a) information granules are the key components of knowledge representation and processing, (b) the level of granularity of information granules (their size, to be more descriptive) becomes crucial to the problem description and an overall strategy of problem solving, (c) there is no universal level of granularity of information; the size of granules is problem-oriented and user dependent.

What has been said so far touched a qualitative aspect of the problem. The challenge is to develop a computing framework within which all these representation and processing endeavors could be formally realized. The common platform emerging within this context comes under the name of Granular Computing. In essence, it is an emerging paradigm of information processing. While we have already noticed a number of important conceptual and computational constructs built in the domain of system modeling, machine learning, image processing, pattern recognition, and data compression in which various abstractions (and ensuing information granules) came into existence, Granular Computing becomes innovative and intellectually proactive in several fundamental ways

- It identifies the essential commonalities between the surprisingly diversified problems and technologies used there, which could be cast into a unified framework we usually refer to as a granular world. This is a fully operational processing entity that interacts with the external world (that could be another granular or numeric world) by collecting necessary granular information and returning the outcomes of the granular computing
- With the emergence of the unified framework of granular processing, we get a better grasp as to the role of interaction between various formalisms and visualize a way in which they communicate.
- It brings together the existing formalisms of set theory (interval analysis), fuzzy sets, rough sets, etc. under the same roof by clearly visualizing that in spite of their visibly distinct underpinnings (and ensuing processing), they exhibit some fundamental commonalities. In this sense, Granular Computing establishes a stimulating environment of synergy between the individual approaches.
- By dwelling on the commonalities of the existing formal approaches, Granular Computing helps build heterogeneous and multifaceted models of processing of information granules by clearly recognizing the orthogonal nature of some of the existing and well established frameworks (say, probability theory coming with its probability density functions and fuzzy sets with their membership functions)

- Granular Computing fully acknowledges a notion of variable granularity whose range could cover detailed numeric entities and very abstract and general information granules. It looks at the aspects of compatibility of such information granules and ensuing communication mechanisms of the granular worlds.
- Interestingly, the inception of information granules is highly motivated. We do not form information granules without reason. Information granules are an evident realization of the fundamental paradigm of abstraction.

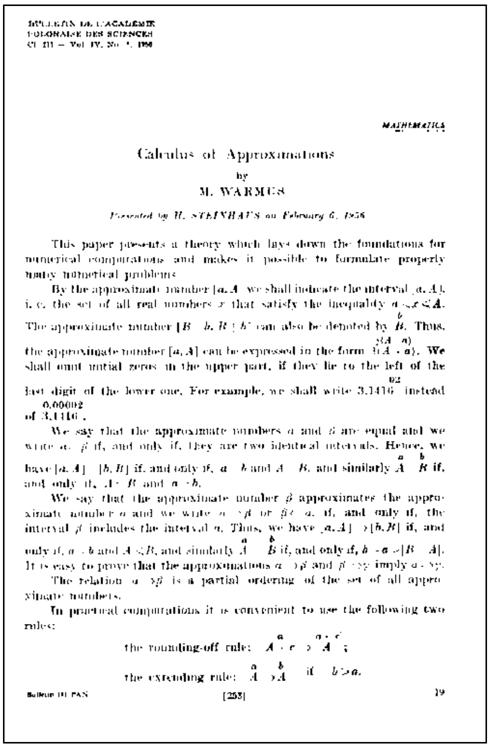
Granular Computing forms a unified conceptual and computing platform. Yet, it directly benefits from the already existing and well-established concepts of information granules formed in the setting of set theory, fuzzy sets, rough sets and others.

2. Formalisms of Granular Computing

As Granular Computing seamlessly embraces and unifies a variety of fundamentals, it is instructive to take a closer look at them and come up with some comparative analysis.

2.1. Interval analysis

The two-valued world of sets and interval analysis ultimately dwells upon a collection of intervals in the line of reals, [a,b], [c,d],…etc. Going back to the history, computing with intervals is intimately linked with the world of digital technology. One of the first papers in this area was published in 1956, see Figure 1. Some other early research was done by Sunaga (1958) and Moore. This was followed by a wave of research in so-called interval mathematics [3][4][9] or interval calculus, cf. http://www.cs.utep.edu/interval-comp. Conceptually, sets (intervals) are rooted in a two-valued logic with their fundamental predicate of membership ($\in$). There holds an important isomorphism between the structure of two-valued logic endowed with truth values (false-true) and set theory with sets fully described by their characteristic functions. The interval analysis is a cornerstone of reliable computing which in turn is ultimately associated with digital computing in which any variable is associated with a finite accuracy (implied by the fixed number of bits used to represent numbers). This limited accuracy gives rise to a certain pattern of propagation of error of computing. For instance, addition of two intervals [a, b] and [c, d] leads to a broader interval in the form [a+c, b+d]. Here the accumulation of uncertainty (or decreased granularity of the result) depends upon the specific algebraic operation completed for given intervals. Interestingly, intervals formed uniformly in a certain space are at the center of any mechanism of analog-to-digital conversion; the higher the number of bits, the finer the intervals and the higher their number. The well known fundamental relationship states that with "n" bits we can build a collection of 2^n intervals of width $(b-a)/2^n$ for the original range of [a,b]. Intervals offer a straightforward mechanism of abstraction: all elements lying within a certain interval become nondistinguishable and therefore are treated as identical. In addition to algebraic manipulation, the area of interval mathematics embraces a wealth of processing including differentiation, integral calculus as well as interval-valued optimization.



BULLETIN DE L'ACADÉMIE
POLONAISE DES SCIENCES
Cl. III — Vol. IV, No. 5, 1956

MATHEMATICS

Calculus of Approximations
by
M. WARMUS

Presented by H. STEINHAUS on February 6, 1956

This paper presents a theory which lays down the foundations for numerical computations and makes it possible to formulate properly many numerical problems.

By the approximate number $[a, A]$ we shall indicate the interval $[a, A]$, i. e. the set of all real numbers x that satisfy the inequality $a \leq x \leq A$. The approximate number $[B - b, B + b]$ can also be denoted by $\overset{b}{B}$. Thus, the approximate number $[a, A]$ can be expressed in the form $\overset{\frac{1}{2}(A-a)}{\frac{1}{2}(A+a)}$. We shall omit initial zeros in the upper part, if they lie to the left of the last digit of the lower one. For example, we shall write $\overset{02}{3.1416}$ instead of $\overset{0.00002}{3.1416}$.

We say that the approximate numbers α and β are equal and we write $\alpha = \beta$ if, and only if, they are two identical intervals. Hence, we have $[a, A] = [b, B]$ if, and only if, $a = b$ and $A = B$, and similarly $\overset{a}{A} = \overset{b}{B}$ if, and only if, $A = B$ and $a = b$.

We say that the approximate number β approximates the approximate number α and we write $\alpha \supset \beta$ or $\beta \subset \alpha$ if, and only if, the interval β includes the interval α. Thus, we have $[a, A] \supset [b, B]$ if, and only if, $a \geq b$ and $A \leq B$, and similarly $\overset{a}{A} \supset \overset{b}{B}$ if, and only if, $b - a \geq |B - A|$. It is easy to prove that the approximations $\alpha \supset \beta$ and $\beta \supset \gamma$ imply $\alpha \supset \gamma$.

The relation $\alpha \supset \beta$ is a partial ordering of the set of all approximate numbers.

In practical computations it is convenient to use the following two rules:

the rounding-off rule: $\overset{a}{A + c} \supset \overset{a+c}{A}$;

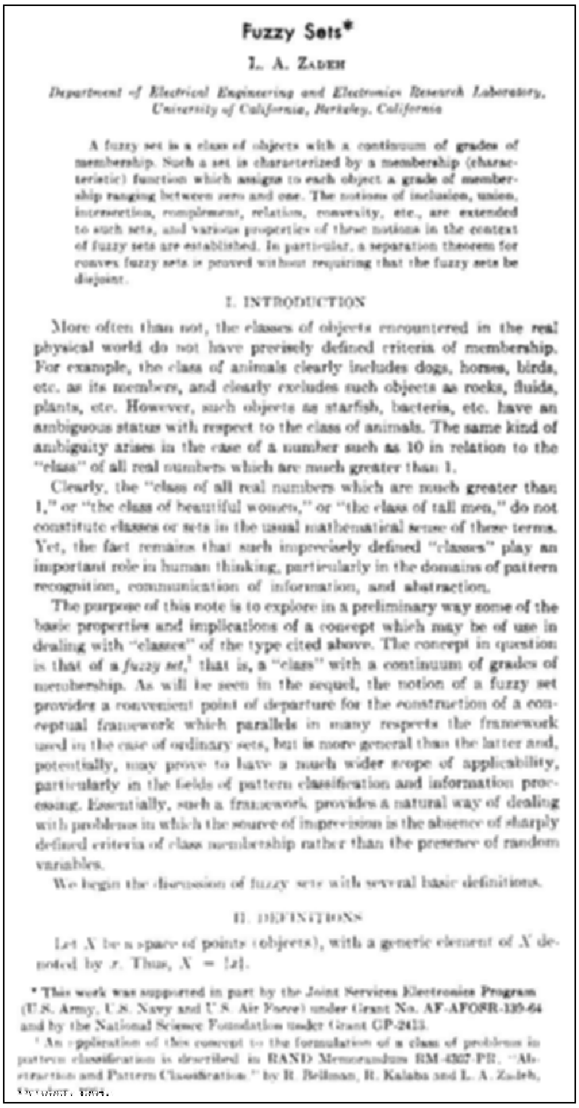
the extending rule: $\overset{a}{A} \supset \overset{b}{A}$ if $b > a$.

Bulletin III PAN [253] 19

Figure 1. Calculus of approximations – at the origin of interval mathematics [18]

2.2. Fuzzy sets

The binary view of the world supported by set theory and two-valued logic has been vigorously challenged by philosophy and logic. The revolutionary step in logic was made by Lukasiewicz with his introduction of three and afterwards multivalued logic [7]. The non-Aristotelian view of the world was vividly promoted by Korzybski [5]. While this sounded like a revolutionary shift of the paradigm at the beginning of the 20th century, the concept of the three valued logic has been broadly embraced later on. For instance, in databases we naturally deal with three logic values by accepting the entries such as "yes", "no", and "don't know". It took however more decades to dwell on the ideas of the non-Aristotelian view of the world before fuzzy sets were introduced, Figure 2. This happened in 1965 with the publication of the seminal paper by Zadeh [19].



Fuzzy Sets*

L. A. Zadeh

Department of Electrical Engineering and Electronics Research Laboratory, University of California, Berkeley, California

A fuzzy set is a class of objects with a continuum of grades of membership. Such a set is characterized by a membership (characteristic) function which assigns to each object a grade of membership ranging between zero and one. The notions of inclusion, union, intersection, complement, relation, convexity, etc., are extended to such sets, and various properties of these notions in the context of fuzzy sets are established. In particular, a separation theorem for convex fuzzy sets is proved without requiring that the fuzzy sets be disjoint.

I. INTRODUCTION

More often than not, the classes of objects encountered in the real physical world do not have precisely defined criteria of membership. For example, the class of animals clearly includes dogs, horses, birds, etc. as its members, and clearly excludes such objects as rocks, fluids, plants, etc. However, such objects as starfish, bacteria, etc. have an ambiguous status with respect to the class of animals. The same kind of ambiguity arises in the case of a number such as 10 in relation to the "class" of all real numbers which are much greater than 1.

Clearly, the "class of all real numbers which are much greater than 1," or "the class of beautiful women," or "the class of tall men," do not constitute classes or sets in the usual mathematical sense of these terms. Yet, the fact remains that such imprecisely defined "classes" play an important role in human thinking, particularly in the domains of pattern recognition, communication of information, and abstraction.

The purpose of this note is to explore in a preliminary way some of the basic properties and implications of a concept which may be of use in dealing with "classes" of the type cited above. The concept in question is that of a *fuzzy set*,[1] that is, a "class" with a continuum of grades of membership. As will be seen in the sequel, the notion of a fuzzy set provides a convenient point of departure for the construction of a conceptual framework which parallels in many respects the framework used in the case of ordinary sets, but is more general than the latter and, potentially, may prove to have a much wider scope of applicability, particularly in the fields of pattern classification and information processing. Essentially, such a framework provides a natural way of dealing with problems in which the source of imprecision is the absence of sharply defined criteria of class membership rather than the presence of random variables.

We begin the discussion of fuzzy sets with several basic definitions.

II. DEFINITIONS

Let X be a space of points (objects), with a generic element of X denoted by x. Thus, $X = \{x\}$.

* This work was supported in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy and U.S. Air Force) under Grant No. AF-AFOSR-139-64 and by the National Science Foundation under Grant GP-2413.

[1] An application of this concept to the formulation of a class of problems in pattern classification is described in RAND Memorandum RM-4307-PR, "Abstraction and Pattern Classification," by R. Bellman, R. Kalaba and L. A. Zadeh, October, 1964.

Figure 2. Fuzzy sets – the next revolution in the non-Aristotelian view at the world [19]

The obvious yet striking difference between sets (intervals) and fuzzy sets lies in the notion of partial membership supported by fuzzy sets. In fuzzy sets, we discriminate between elements that are "typical" to the concept and those of borderline character. Information granules such as *high* speed, *warm* weather, *fast* car are examples of information granules falling under this category can be conveniently represented by fuzzy sets. As we cannot specify a single, well-defined element that forms a solid border between full belongingness and full exclusion, fuzzy sets offer an appealing alternative and practical solution to this problem. Fuzzy sets with their smooth transition boundaries form an ideal vehicle to capture the notion of partial membership. In this sense information granules formalized in the language of fuzzy sets support a vast array of human-centric pursuits. They are predisposed to play a vital role when interfacing human to intelligent systems.

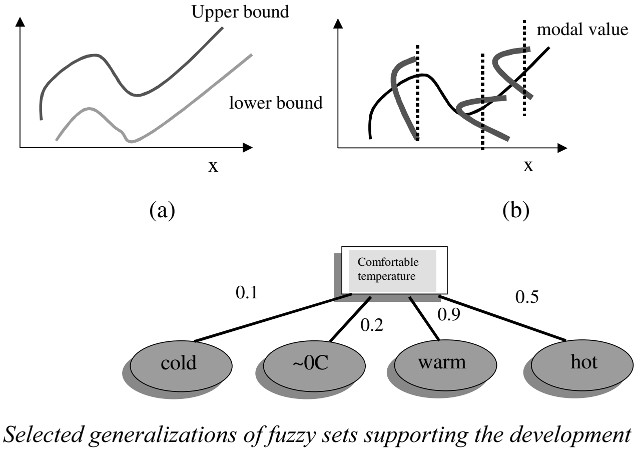
From the optimization standpoint, the properties of continuity and commonly encountered differentiability of the membership functions are a genuine asset. We may easily envision situations where those information granules incorporated as a part of the entire neurofuzzy system are subject to optimization – hence the differentiability becomes of critical relevance. What becomes equally important is the fact that fuzzy sets bridge numeric and symbolic concepts.

Since the inception of fuzzy sets, observed was a rapid progress along the line of enhancing their capabilities of knowledge representation, see Figure 2. This gave rise to various highly appealing concepts such as e.g.,

(a) interval-valued fuzzy sets; here we admit membership grades to form numeric intervals in the unit interval,

(b) fuzzy sets or type-2 with the membership modeled by fuzzy sets defined in the unit range of the membership grades [8], and

(c) fuzzy sets of order-2 using which we model composite concepts whose components are fuzzy sets themselves. For instance, the idea of a *comfortable* temperature is formed on a basis of more generic concepts (again represented by fuzzy sets).



(a) (b)

Figure 3. Selected generalizations of fuzzy sets supporting the development of higher, conceptually rich concepts: (a) interval-valued fuzzy sets, (b) type-2 fuzzy sets, (c) order-2 fuzzy sets

By introducing a certain α-cut, we can transform a fuzzy set into a set. By choosing this threshold level (α), we admit elements whose membership grades are deemed meaningful (from the standpoint of the imposed threshold) and positioned above this threshold value. This could lead to a deceptive impression that any fuzzy set could be made equivalent to some set. In essence, by building any α-cut we elevate some membership grades to 1 (full membership) and eliminate others with lower membership grades (total exclusion). Surprisingly, no account is taken for the distribution of elements with partial membership so that this effect cannot be quantified in the resulting construct of the α-cut. The idea of shadowed sets [13][15] is aimed at alleviating this problem by constructing regions of complete ignorance about membership grades. In essence, a shadowed set $A^{\sim}$ induced by given fuzzy set A defined in **X** is a mapping such that $A^{\sim}$: **X** → { 0, 1, [0,1]}. Given $A^{\sim}(x)$, the two numeric values that is 0 while 1 take on a standard interpretation: 0 denotes complete exclusion from $A^{\sim}$, 1 stands for its complete inclusion in A. The value of $A^{\sim}(x)$ equal to [0,1] represents a complete ignorance – we conclude that nothing is known about membership of x in $A^{\sim}$: we *neither* confirm its belongingness to $A^{\sim}$ *nor* commit to its exclusion. In this sense, such "x" is the most "questionable" point and should be treated as such (e.g., this could trigger some action to analyze this element in more detail, exclude it from further analysis, etc.). The name of shadowed set is a descriptive reflection of a set which comes with "shadows" positioned around the edges of the characteristic function, Figure 4.

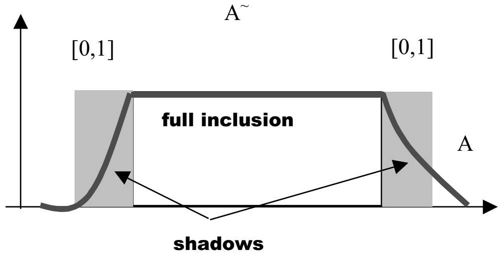
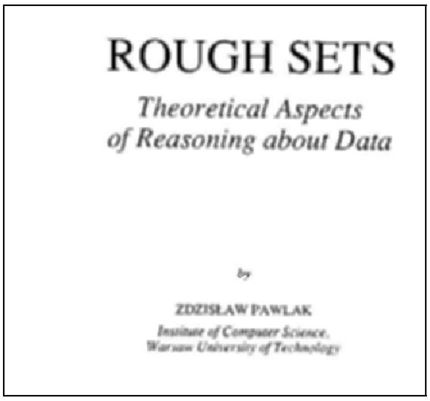


Figure 4. From fuzzy set A to its three-valued interpretation in the form of induced shadowed set $A^{\sim}$; observe that "shadows"are produced at the edges of the characteristic function reflecting some intermediate membership values of the fuzzy set A

2.3. Rough sets

The description of information granules completed with the aid of some vocabulary is usually imprecise. Intuitively, such description may lead to some approximations, called lower and upper bounds. This is the crux of rough sets introduced by Pawlak [11], Figure 5. Interesting generalizations and insights are offered in [6][10]16][17].



ROUGH SETS

Theoretical Aspects of Reasoning about Data

by

ZDZISŁAW PAWLAK

Institute of Computer Science,
Warsaw University of Technology

Figure 5. Rough sets –new models of information granules with rough boundaries [12]

As visualized in Figure 6, we are concerned with a description of a given concept X realized in the language of a certain collection (vocabulary) of rather generic and simple terms A_1, A_2, …, A_c. The lower and upper boundaries (approximation) are reflective of the resulting imprecision caused by the conceptual incompatibilities between the concept itself and the existing vocabulary.

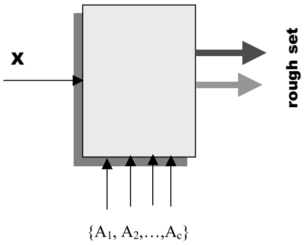


Figure 6. Rough set as a result of describing X in terms of some fixed vocabulary {A_1, A_2, .,A $_c$}

To illustrate the effect of forming the approximations, let us consider a concept of acceptable temperature treated as an interval and assuming values in the range of [a, b]. We intend to describe it using the vocabulary of terms being a collection of the uniformly distributed intervals as visualized in Figure 7. Apparently the concept (shown in solid thick line) to be described fully "covers" (includes) one interval that is I_3. There are also some intervals that have at least some limited overlap with it and those are I_2, I_3, and I_4. In other words, we say that the concept when characterized by the predefined vocabulary of intervals does not lend itself to a unique description and the best characterization we can deliver comes in the form of some boundaries (approximations). The tighter the approximations, the more detailed the description.

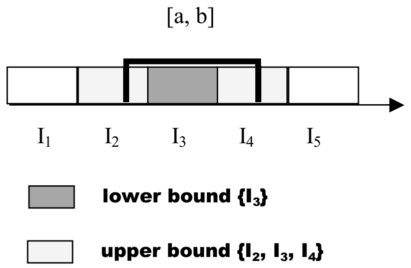


Figure 7. Concept (set) [a, b] represented in the language of uniformly distributed intervals; note the upper and lower bound forming their representation

Quite often rough sets and fuzzy sets form some hybrids. Then we usually talk about rough-fuzzy sets or fuzzy-rough sets [2].

In general, the hybridization effect between different fundamentals of information granules occurs quite often. Not only we could see this in the case of rough-fuzzy sets. The orthogonality of probability (and probabilistic granules) and fuzzy sets produces an interesting and practically relevant effect of combining those two in the form of probabilities of fuzzy event or fuzzily quantified probabilities. There are also constructs such as fuzzy random variables and probabilistic sets.

3. The Development of Information Granules

As information granules form generic building blocks of granular models and are a subject of all processing, their efficient development is of paramount importance to any successful application. In a nutshell, we can distinguish between three main categories of approaches such as (a) user-driven, (b) data-driven, and (c) hybrid. This taxonomy is driven by the methodology and techniques we use to incorporate domain knowledge about the information granules and elicit their further detailed description.

The user-based approach, as the name suggests, is concerned with forming information granules on a basis of some information available from experts, users, and developers of the system under consideration. In common, it is regarded that the elicitation of characteristic or membership functions is reflective of a way in which we abstract the concepts. As information granules are human-centric, the human factor is the essential source of knowledge that nobody can argue about. Perhaps this is true to a significant extent given the fact that information granules do not exist as *physical* entities that could be easily measured (such as length, current, voltage, etc.) but emerge as concepts and useful abstractions. The user driven approach was often challenged, particularly in the context of fuzzy sets (where are the membership functions coming from? - this was a typical question one commonly encountered in the past).

The data-driven approach to information granulation exploits data as a sole source of knowledge about the underlying information granules. In essence, we could view this as a transformation of a large number of data points into a very limited number of their abstractions – information granules. Various clustering techniques are useful here. Depending upon the category of the clustering algorithm, we exploit a certain category of the formalism of information granulation. For instance, K-means develops a collection of set-based information granules. Fuzzy C-Means (FCM), on the other hand, leads to a collection of fuzzy sets (fuzzy relations). In data-driven approach, information granules are a result of some optimization procedure. Their membership functions depend on the values of the parameters of the clustering technique; a typical example here would be a fuzzification coefficient (factor) used in the FCM.

The user-driven approach seems to be completely disjoint from the data-oriented one. It is true to some extent. The approach does not take data directly into account. We should be cognizant however that any time we form a certain information granule, its emergence is *implicitly* affected by our previous experience and hence the data we were continuously exposed to in the past.

The hybrid approach is positioned somewhere in-between these two extremes. As the name implies, we rely on data in the design of the information granules (so clustering techniques are of interest here) however such clustering mechanisms are augmented and enhanced by some hints coming from the designer/expert. Given the augmented format of the clustering, it is referred to as a so-called knowledge-based clustering; the term being coined in [15]. The knowledge-based enhancements could come in the form of some labeled data (usually there is a small percentage of those), proximity hints (where the user and/or designer conveys some referential type of guidance by expressing a level of proximity between some pairs of data), uncertainty guidelines (where patterns are labeled with respect to the level of confidence as to their assignment to clusters), etc. These guidelines usually augment the original objective function used in the FCM and thus modify the conditions under which the information granules are discovered (optimized).

4. Quantifying Granularity: Generality Versus Specificity

The notion of granularity itself and a level of specificity/generality seem to be highly intuitive: we can easily sense what is more detailed and specific and what looks more abstract and general. Formally, we can easily quantify granularity by counting the number of elements captured by the given information granule. The more elements, the lower the granularity. In limit, a single element exhibits the highest level of granularity (specificity). In the case of sets, this will be the cardinality (number of elements) or the length of the interval or a similar measure. In case of fuzzy sets, we usually use a so-called sigma-count that is produced by summing up the membership grades of the elements belonging to the fuzzy set under consideration. For rough sets, we may consider the cardinality of their lower or upper approximations.

Practically, there is no universal and uniquely acceptable level of granularity of information. The level inherently depends on the problem at hand, intention of the user, and the needs of the customer. Time could be granulated in nanoseconds. It could also be granulated in years. Which granulation is better? It depends. If you are a circuit designer, it is likely that the first granulation is the most suitable. When discussing planning of some manufacturing activities, the second granulation sounds just right.

5. Communication between Systems of Information Granules

Information granulation and resulting information granules form a certain conceptual framework within which all processing activities take place [1][15]. As already emphasized, the selection of information granules, their size and specific formal representation framework are very much problem dependent issues. The flexibility we are provided in this manner is essential to the efficient problem representation and problem solving. The size of information granules helps us capture the level of required specificity of the solution to the problem. The design decision about the formal setting is essential and is made based on the available domain knowledge, expertise of the design team, format of experimental numeric or granular data and expected outcomes of the modeling activities (that could result in the form of some control., classification, or decision-making algorithms). As there could be a variety of computing processing going on at the same time with some communication, we can envision a distributed model in which each process can be treated individually, Figure 8.

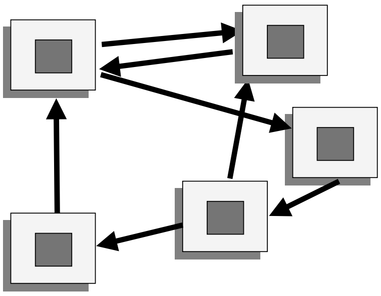


Figure 8. Granular Computing in the realization of autonomous processes (agents); distinguished are their processing cores (denoted by darker color) and communication layers.

Each agent operates in its environment of granular computing and communicates with others. From the high-end architectural standpoint, we can envision two-layer architectures with some core supporting all computing with some communication layer surrounding it. Computing processes are realized in the framework set up via some formalism of Granular Computing. For instance, there could be an agent operating on fuzzy sets and fuzzy relations. For the other one the computing framework results in interval calculations. The role of the communication layer is to facilitate acceptance of data and support exchange of results produced by other agents. As the heterogeneity of information granulation is profound, the functional demands on this layer are immense. Firstly, one has to assure communication between the same formal frameworks whose information granules could exhibit a very different level of granularity (specificity), Figure 9. In other words, we should be able to handle specificity (abstraction) incompatibilities. Secondly, there should be mechanisms that support exchange and translation of information granules formed within different formal settings (intervals, fuzzy sets, rough sets, …).

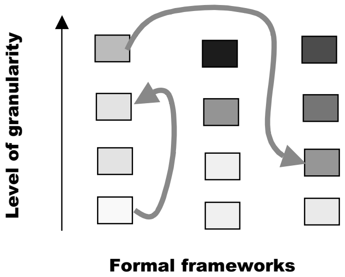


Figure 9. Facets of communications between systems operating at different levels of information granularity and using different formal frameworks of information granulation; hence distinguished are mechanisms of vertical and horizontal communication

With regard to the vertical communication (the one where we are within the same formal framework of information granules but encounter different levels of granularity), the underlying issue is how to facilitate such communication and when the communication could become infeasible because of the significant differences in the levels of information granularity of the collaborating environments. When it comes to the horizontal communication, it requires transformation of granules between two formal frameworks. This, in general, is more complicated and there are a number of possible alternatives. For instance, when dealing with communication between fuzzy sets and intervals, one may consider α-cuts, which transform fuzzy sets into the corresponding two-valued counterpart. When communicating between the framework of fuzzy sets and probabilities, a so-called probability-possibility transformation is in use.

In the context of communication, it is worthwhile to emphasize that such mechanisms are applicable in the limit condition when a granular environment interacts with a continuous (analog) world. We point here at two prominent examples, Figure 10, which again help us appreciate that unified treatment of concepts as being offered by Granular Computing.

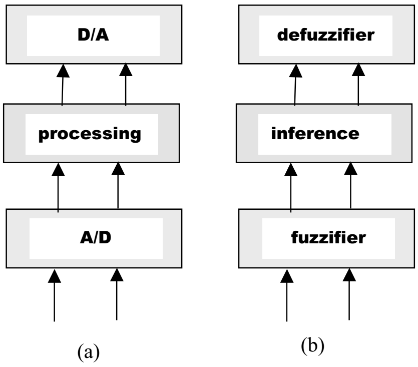


Figure 10. Communication between continuous world and digital processing (a) and fuzzy computing (b)

In digital world (which hinges on the concepts of interval calculus), we are faced with the communication with the analog world. Evidently, analog-to-digital (A/D) and digital-to-analog (D/A) converters play a pivotal role. A certain communication mechanism is also required in the case of fuzzy computing; here we use the terms of fuzzifier and defuzzifier. In spite of this different nomenclature that is typical to fuzzy sets, the role of such functional modules is identical to those played by A/D and D/A converters: we interact with the external world by accepting information or releasing the results of processing into the environment. In both cases, the design problem concerns a minimization of the possible distortion occurring during such communication.

Organizing the components of Granular Computing discussed so far, we can arrive at the formal concept of a granular world being treated as a cohesive conceptual and computational entity being expressed as the following four-tuple

< X , F, F, C>

with **X** forming a certain universe of discourse, **F** denoting a certain formal framework of granulation, **A** being a collection of generic information granules (that imply an already accepted level of information granularity), and **C** describing the relevant communication mechanisms.

6. Granular Computing and Computational Intelligence

Granular Computing seamlessly integrates with architectures of Computational Intelligence. Given the fact that information granules help set up the most suitable perspective when dealing with the problem, collecting data (that could be of heterogeneous character), carrying out processing and releasing the results (in a formal acceptable to the environment), the general architecture is shown in Figure 11.

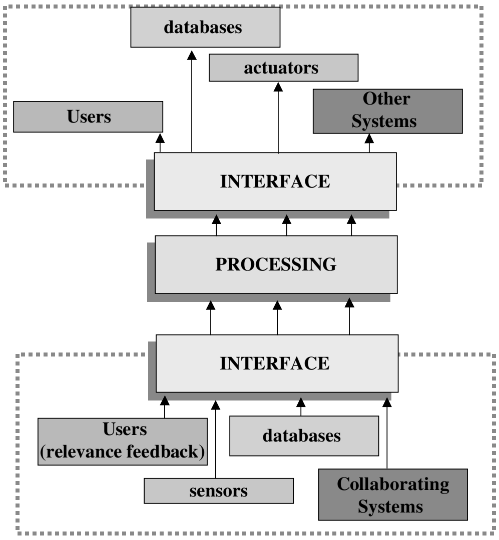


Figure 11. The architecture of systems of Computational Intelligence with the functional layers of communication (interfacing) with the environment and the processing core

While the communication layers are supported by Granular Computing, the underlying processing is a domain of neurocomputing while the overall optimization of the architecture is supported by the machinery of evolutionary computing. There are different levels of synergy; for instance, one could regard the overall architecture as a neurofuzzy system. In this case the interface delivers a unified setting where various sources of data are effortlessly combined and presented to the neural network, which constitutes the core of the processing layer. In many cases, the architecture could have somewhat blurred delineation between the communication layers and the processing core, in particular when information granules become an integral part of the basic processing elements. A typical example here comes when we are concerned with a granular neuron – a construct in which the connections are treated as information granules, say fuzzy sets (and then we may refer to it as a fuzzy neuron) or rough sets (which gives rise to rough neurons).

As discussed earlier, information granules help us cast the problem into some perspective. This becomes visible when in case of neural networks. To cope with huge masses of data, we could granulate them (which naturally reduces their number and dimensionality) and treat those as meaningful aggregates and components of the learning set. Obviously, we should become aware of the fact of possible communication aspects this granulation brings into the picture and assess their feasibility.

7. Conclusions

Granular Computing has emerged as a new paradigm that builds a comprehensive and well-rounded conceptual framework being endowed with well-established individual underlying theories and algorithms. It is an important functional component of Computational Intelligence with the agenda that synergistically complements various aspects of learning and optimization.

The role of Granular Computing in the design of intelligent systems could be considered from two different standpoints. First, it is a systematic development framework with solid fundamental underpinning as far as the design principles are concerned (addressing such fundamentals as e.g., abstraction, hierarchical and stepwise development and refinement). The diversity of theoretical fundamentals helps us cope with and integrate various formats of granular information. Second, the algorithmic maturity of each of the main formalisms and related technologies makes Granular Computing an attractive development vehicle.

Even though venturing into the future could be somewhat risky, one could offer some general observations. We could envision several interesting trends in the development of Granular Computing regarded as a standalone discipline as well as the integral component of Computational Intelligence. There will be a growing interest in the hybridization of various formal frameworks of Granular Computing. While the ideas of fuzzy probabilities, fuzzy rough sets and alike have been attractive in the past, we will eventually witness more studies along this line capitalizing on the orthogonality of various underlying formal models. This in essence will support further knowledge integration completed in the framework of Granular Computing. As the entire discipline of Computational Intelligence is steadily moving towards user-centric systems, Granular Computing will be vital to its further developments.

References

1. A.Bargiela, W. Pedrycz, *Granular Computing: An Introduction*, Kluwer Academic Publishers, Dordercht, 2003.
2. D. Dubois, H. Prade, Rough fuzzy sets and fuzzy rough sets, *Int. J. General Systems* 17, 2–3, 1990, 191–209.
3. E. Hansen, A generalized interval arithmetic, *Lecture Notes in Computer Science*, Springer Verlag, vol. 29, 1975, 7-18.
4. L. Jaulin, M. Kieffer, O. Didrit, E. Walter, *Applied Interval Analysis*, Springer, London, 2001.
5. A. Korzybski, *Science and Sanity: An Introduction to Non-Aristotelian Systems and General Semantics*. 3rd ed. The International Non-Aristotelian Library Publishing Co., Lakeville, C.T., 1933.
6. T.Y. Lin, Data mining and machine oriented modeling: a granular computing approach, *J. of Applied Intelligence*, 13, 2, 2000, 113-124.

7. J. Lukasiewicz, Philosophische Bemerkungen zu mehrwertigen Systemen des Aussagenkalk, *C. R. Soc. Sci. Lettres de Varsovie*, 23, 1930, 51-77.
8. J.M. Mendel, On a 50% savings in the computation of the centroid of a symmetrical interval type-2 fuzzy set , *Information Sciences*, In press, Available online 2 July 2004.
9. R. Moore, *Interval Analysis*, Prentice Hall, Englewood Cliffs, NJ, 1966.
10. S.K. Pal, A. Skowron (eds.), *Rough Fuzzy Hybridization. A New trend in Decision-Making*, Springer Verlag, Singapore, 1999.
11. Z. Pawlak, Rough sets, *Int. J. Comput. Inform. Sci.* 11, 1982, 341–356.
12. Z. Pawlak, *Rough Sets. Theoretical Aspects of Reasoning About Data*, Kluwer Academic Publishers, Dordercht, 1991.
13. W. Pedrycz, Shadowed sets: representing and processing fuzzy sets, *IEEE Trans. on Systems, Man, and Cybernetics, part B*, 28, 1998, 103-109.
14. W. Pedrycz (ed.), *Granular Computing: An Emerging Paradigm*, Physica Verlag, Heidelberg, 2001.
15. W. Pedrycz, *Knowledge-based Clustering*, J. Wiley, 2005.
16. L. Polkowski, A. Skowron (eds.), *Rough Sets in Knowledge Discovery*, Physica Verlag, Heidelberg, 1998.
17. A. Skowron, Rough decision problems in information systems, *Bulletin de l'Academie Polonaise des Sciences (Tech)*, 37, 1989, 59-66.
18. M. Warmus, Calculus of approximations, *Bulletin de l'Academie Polonaise des Sciences*, 4, 5, 1956, 253-259.
19. L.A. Zadeh, Fuzzy sets, *Information & Control*, 8, 1965, 338-353.
20. L.A. Zadeh, Fuzzy sets and information granularity, In: M.M. Gupta, R.K. Ragade, R.R. Yager, (eds.), *Advances in Fuzzy Set Theory and Applications*, North Holland, Amsterdam, 1979, 3-18.
21. L.A. Zadeh, Fuzzy logic = Computing with words, *IEEE Trans. on Fuzzy Systems*, 4, 1996, 103-111.
22. L.A. Zadeh, Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic, *Fuzzy Sets and Systems*, 90, 1997, 111-117.
23. L.A. Zadeh, From computing with numbers to computing with words-from manipulation of measurements to manipulation of perceptions, *IEEE Trans. on Circuits and Systems*, 45, 1999, 105-119.

Part II

Classification and Clustering

Parallel Neuro Classifier for Weld Defect Classification

S V Barai and Piyush Agrawal

Department of Civil Engineering
Indian Institute of Technology Kharagpur
Kharagpur, India

Abstract. It is of utmost important to maintain perfect condition of complex welded structures such as pressure vessels, load bearing structural members and power plants. The commonly used approach is non-destructive evaluation (NDE) of such welded structures. This paper presents an application of artificial neural networks (ANN) for weld data, extracted from reported radiographic images. Linear Vector Quantization based supervised neural network classifier is implemented in Parallel Processing Environment on PARAM 10000. Single Architecture Single Processor and Single Architecture Multiple Processor based parallel neuro classifier are developed for the weld defect classification. The results obtained for various statistical evaluation methods showed promising future of Single Architecture Single Processor based parallel neuro classifier in the problem domain.

Keywords: Classifier, Linear Vector Quantization, neural networks, parallel processing, weld defect

1 Introduction

The assessment of the safety and reliability of existing welded structures such as pressure vessels, load bearing structural members and power plants, has been the focus of much investigation in recent years. An assessment of a welded structural system requires quantitative and qualitative data concerning the current state of the structure, its strength and response characteristics, and a methodology to integrate various types of information into a decision-making process to evaluate the safety of entire structure. Perhaps the most challenging aspect of weld evaluation to develop a rational methodology to synthesize the diverse information related to the structural weld condition and its behavior.

In practice, non-destructive evaluation (NDE) technologies have been used very often for weld evaluation (Berger 1977, Bray and Stanley 1989). In a broad sense, NDE can be viewed as the methodology used to assess the integrity of the structure without compromising its performance. Recently, many studies have reported results where signal processing and neural networks (NN) were used in characterizing defects of weld based using NDE data (Rao et al. 2002, Liao and Tang 1997, Nafaa et al. 2000, Stepinski and Lingvall 2000).

Radiographic testing is one of the most popular NDE techniques adopted in inspecting welded joints. Usually real-time radiographic weld images are produced during radiographic testing of welded component (Bray and Stanley 1989). These images are digitized without losing important information. Application of feature extraction methods to such digitized images helps in identifying features (Liao and Ni 1996, Liao and Li 1998). Further, Liao and his research group have proposed detection of welding flaws from radiographic images using soft-computing tools such as fuzzy clustering method and fuzzy k-nearest neighbor algorithms (Liao et al. 1999, Liao and Li 1997).

Advancement in the field of data mining (Fayyad et al. 1996) can help researchers handle complex problems like classification where many features extracted from digitized radiographic images play an important role. Liao et al. (2001) have attempted to explore data mining approach for weld quality models, constructed with multiplayer perceptron networks. Recently, Barai and Reich (2002) had proposed data mining approach for weld defect classification using Linear Vector Quantization (LVQ) neural networks model. In the present study, LVQ neural networks model is selected due to reported excellent performance.

In the course of weld defect data modeling, many neural network models could be created at the cost of computing time. Sometimes it may take a few days to train a network for a particular engineering problem. Such networks are usually modeled using conservative general guidelines. Instead, neural network models could be selected for a particular task, based on statistical tests.

The principal motivation to using a parallel computing environment is to reduce the time required to solve a computational problem. Parallel computing takes place when the processing task is divided among the processors available on a network. These processors work in parallel and in cooperation to solve a complex problem. As artificial neural network (ANN) applications move from laboratories to the practical world, the concern about reducing the time for learning and recall phases become very real. Parallel implementation schemes offer an attractive way of speeding up both these phases of artificial neural networks (Labonte and Quintin 1999, Seiffert 2002).

The present research work is an attempt to solve the problem of weld defect data modeling by multiple *neural network schemes* on PARAM 10000 (refer Appendix). The developed tool can be implemented for various kinds of engineering applications. In the present paper, following objectives are identified.

- Develop Linear Vector Quantization (LVQ) based supervised neural network classifier in parallel processing environment.
- Implement parallel neuro classifiers - Single Architecture Single Processor and Single Architecture Multiple Processor on PARAM 10000.
- Demonstrate developed neuro classifiers for the weld defect classification problem and carry out performance study.

2 Neural Networks

The following section describes about the selection of classifier and its evaluation of classifier.

2.1 Selection of Classifier

Neural networks are usually employed to classify patterns based on learning from examples. In the literature, varieties of neural network models, such as Hopfield net, Hamming net, Carpenter/Grossberg net, single-layer perceptron, multilayer network etc., are available. The single-layer Hopfield and Hamming nets are normally used with binary input and output under supervised learning. The Carpenter/Grossberg net, however, implements unsupervised learning. The single-layer perceptron can be used with multi-value input and output in addition to binary data. A serious disadvantage of the single-layer network is that complex decision may not be possible. The decision regions of a single-layer network are bound by hyperplanes whereas those of two-layer networks may have open or closed convex decision regions (Haykin 1994, Lippman 1987). One can select the model depending upon the application domain. The multilayer network is a very popular artificial neural network architecture and has performed well in a variety of applications in several domains including classification from radiographic testing (Liao and Tang 1997, Stepinski and Lingvall 2000).

Kohonen rule based Linear Vector Quantization algorithm was used by Barai and Reich (2002). Learning vector quantization (LVQ) is a method for training competitive layers in a supervised mode (Kohonen, 1995). A competitive layer will automatically learn to classify input vectors. However, the classes that the competitive layer finds are dependent only on the distance between input vectors. If two input vectors are very similar, the competitive layer probably will put them into the same class. For the present study, LVQ algorithm is selected as a classifier (Kohonen, 1995).

2.2 Classifier Performance Evaluation Methods

Various issues related to network performance evaluation are discussed elsewhere (Reich and Barai 1999). Typical NN model evaluation methods are: (a) Resubstitution (b) Split Sample Validation / Hold-out Validation (c) Cross-Validation such as K-fold cross validation, Leave-one-out method (Reich 1997, Reich and Barai 2000). These methods have been used to evaluate the performance of model for the problem domain.

3 LVQ Implementation on PARAM 10000

Parallel programming in PARAM 10000 is carried out using the Message Passing Library. Message Passing Interface (MPI) was developed in 1993 – 1994 by a group of researchers (http://www.mpi-forum.org/). It is one of the best standards for programming parallel processors and is based on message passing. MPI is a standard specification for message passing libraries. MPI makes it relatively easy to write portable parallel programs. The main advantages of establishing a message-passing interface for such environments are portability and ease of use. And a standard memory-passing interface is a key component in building a concurrent computing environment.

3.1 Single Architecture Single Processor

There are total 8 processors available in PARAM 10000 with ranks from 0 to 7. Processor with rank 0 acts as the Master Processor and rest as Slave Processors. The job of Master Processor is to control all other processors and assign them their jobs. Once the program is executed Master Processor reads the required information from the files and assigns a job to each slave processors. Once a processor completes a job, it will notify the master and a new job is assigned to it. This process is continued till the entire job is finished. In this way no processor remains idle. So, the available resources are optimally utilized. Finally, outputs generated by different processors are stored. Algorithm of Single Architecture Single Processor on PARAM 10000 is given below.

Step 1: Before any other MPI functions can be called, the function MPI_Init must be called. It allows systems to do any special set up so that the MPI library can be used. Calling MPI_Comm_rank and MPI_Comm_size respectively returns rank of the processor and the number of processor.

Step 2: If the rank of processor is ‘0’ i.e. if the processor is Master Processor, it assigns a job to each of the processor and waits to get a finish signal from other processors.

Step 3: All processor except Master Processor read the Input file containing all the examples, randomize it and write it in a new file.

Step 4: From a file having the information about the architecture following values are read and memory is allocated accordingly. (a) Input units (b) Kohonen rows and columns (c) Output units (d) Training epochs (e) Number of examples

Step 5: From a limit file containing the (a) Maximum and minimum values of each feature among all the examples and (b) Percentage of each class in the whole data set are read. The weight matrix of the Kohonen layer is assigned based on the maximum and minimum values read above. The weights of output layers are assigned according to the percentage of class.

Step 6: After all the weight matrices are initialized, training is started. One input example at a time is read from the input file created in step 4. Then the values of learning rate and width of topological neighborhood function are calculated based on iteration step. After all the values are known, training is performed based on the LVQ algorithm and weights are updated. Number of iterations is different in different neuro models. (a) Resubstitution: Whole data set is used for training. (b) Hold-Out: 67 percent of the data examples is used for training. (c) K-fold: 90% of the data examples is used for training.

Step 7: Once the network is properly trained, evaluation of the network is performed based on various evaluation methods as discussed earlier. They are - (a) Resubstitution: Input file created in step 4 is read again and network is tested for the whole data examples. (b) Hold-Out: In step 7 only 67% of data were read. The same file is continued and network is tested for the remaining 33% of data. (c) K-fold: Trained network is tested for the remaining 10% of data. There is a difference in the testing and training procedure in K-fold method. Once the randomized input file is created in step 4, it is divided into 10 subsets and each time leaving out one of the subsets from training and that subset is used for the testing. This process is repeated 10 times as 10 different training and testing data sets are generated for a given input file.

Step 8: After the completion of simulation, the architecture values and weights of Kohonen layer and output layer are written to an output file.

Step 9: The total time taken by the processor to execute step 3 to step 8 is printed on the screen.

Step 10: The processor reports to the Master Processor about the completion of job. If some job is still pending, it is assigned to the processor or a signal to quit is sent.

Step 11: As soon as the program has finished using the MPI library, it calls MPI_Finalize function to clean up any unfinished task, if left by MPI.

3.2 Single Architecture Multiple Processor

In this method total number of neurons is distributed among the available processors. As in *Single Architecture Single Processor* Master Processor controls the flow of program. It distributes neurons among the processors and also calculates the global winner. Algorithm of Single Architecture Multiple Processor on PARAM 10000 is given below.

Step 1: Before any other MPI functions can be called, the function MPI_Init must be called. It allows systems to do any special set up so that the MPI library can be used. Calling MPI_Comm_rank and MPI_Comm_size respectively returns rank of the processor and the number of processor.

Step 2: From a file having information about the architecture, following values are read and memory is allocated accordingly. (a) Input units (b) Kohonen rows and columns (c) Output units (d) Training epochs (e) Number of examples

Step 3: If the rank of processor is ‘0’ i.e. if the processor is Master Processor, it distributes the available number of units among other processors, based on the size of Kohonen layer and total number of processors available. It also sends the position of the starting neuron in the two dimensional lattice of the Kohonen layer to each processor, as absolute position of a neuron is required while calculating the lateral distance between winning neuron and excited neuron in the topological neighborhood function.

Step 4: From a limit file, the (a) Maximum and minimum values of each feature among all the examples and (b) Percentage of each class in the whole data set are read. The weight matrix of the Kohonen layer is assigned based on the maximum and minimum values read above. The weights of output layer are assigned according to the percentage of class

Step 5: After all the weight matrices are assigned, training is started. One input example at a time is read from the input file. Then the values of learning rate and width of topological neighborhood function are calculated based on iteration step. After all the values are known, local winner is calculated by all slave processors based on the LVQ algorithm and its absolute position and value is returned to the Master Processor.

Step 6: Once Master Processor gets all local winner and its value, it calculates the global winner and broadcasts it to all the slave processors.

Step 7: Then all the slave processors update its weights according to the LVQ algorithm.

Step 8: Step 5 to step 8 is repeated for all the examples and a completion signal is sent to the Master Processor.

Step 9: Then Master Processor calls each processor in sequence to write its weight matrix in the same output file.

Step 10: As soon as the program has finished using the MPI library, it calls MPI_Finalize function to clean up any unfinished task, if left by MPI.

Step 11: Depending upon the size of problem and the time required to train and validate a network, (a) either single network is trained in single processor, enabling us to train many architectures in parallel or (b) single network is trained in all the available processors by distributing the neurons in a layer among the processors, enabling us to handle a very large problem.

4 Data Acquisition

Non-destructive testing (NDT) of welded structure is used very often for failure analysis of important structures. Radiographic testing is one of the most popular NDT techniques adopted in inspecting welding joints. Usually real-time radiographic weld images are produced during the radiographic testing of welded component. Liao and Tang (1997) collected X-ray strips of about 3.5 inches wide by 17 inches long. They were digitized at 70 μm resolution. These digitized images were produced using 5000 pixels by 6000 lines images. From these images down-sampled image of size 250 pixels by 300 lines were produced to find anomalies in weld. The down-sampled images were used for weld extraction. In order to formulate the classification problem of weld from non-welds, feature extraction was essential. Three features were defined for each object in line image and they are as follows.

- The width **(x_1)**
- The mean square error between the object and its Gaussian intensity plot **(x_2)**
- The peak intensity **(x_3)**

Feature space is generally of a much lower dimension than the original data space. The feature extraction algorithm consists of three major steps:

- A trough detection algorithm is applied to find the troughs associated with each object detected. Once the troughs are found, the width of the object can be calculated.
- The Mean Square Error between the intensity profile of the object and its Gaussian is computed.
- A peak detection algorithm is used to detect objects in a line image. For each detected object, the peak intensity and the peak position are obtained. The peak position is not used to identify but to find the weld after it is identified.

Liao and his research group extracted the samples that contain linear and non-linear welds. They obtained a total of 2493 samples. One feature of example number 1399 was missing in the original data set. Nearest neighbor algorithm is used to find the missing information for the present study. This algorithm finds the nearest match of the missing example with some other examples belonging to the same class as the missing example. Thus the missing information was obtained in present study. In this present investigation, neural networks will be trained to identify whether the patterns are welds or non-welds.

5 Neural Networks Modeling for Weld Classification

5.1 Input and Output Parameters

There are three inputs to the neural network. These are the three features extracted from the radiographic images of the welds. They are: Width of the object (x_1), Mean Square Error between the object and its Gaussian (x_2)

and the peak intensity or gray level of the object (x_3). Output to the network is the class to which the particular sample belongs. There are two classes having values either '1 'or '0'. '1' is used to signify that weld is proper. '0' stands for improper weld.

5.2 Neural Network Architecture and Training

In the Liner Vector Quantization only three layers are possible, namely, Input layer, Competitive Layer (Kohonen layer) and Output Layer.

- *Input Layer:* As three features are extracted from the radiographic images of the welds, so the number of neurons in the Input Layer is three. Hence, input layer is a vector of size 3. Next is the Competitive Layer.
- *Competitive Layer:* This is a two dimensional layer. To decide the number of neurons in this layer, one should consider the overall performance of the network. Going through different hit and trial, and on the basis of experience and compromise between accuracy and computational time, itt is decided to have 144 neurons. These neurons are distributed uniformly in 12 rows and 12 columns. So, the size of this layer is 12 by 12.
- *Output Layer:* This is a two dimensional layer. Only two outputs are possible because the network has to classify the input examples into two classes. So, the number of rows in the output layer is two. The number of columns is equal to the number of neurons in the Competitive Layer, which is equal to 144. The weights in this layer are pre assigned depending upon the data set. In this model there are 2493 examples and almost 50% examples belong to class '0' and 50% belong to class '1'. So, neurons from 1 to 72 are assigned to class '0' and rest to class '1'.

The training exercises are carried out using developed parallel neuro classifiers. Number of epochs used is 100, which is decided by maintaining a balance between total time and the error. A low value of .05 is used for learning rate so that network will learn properly. Initial width of topological neighborhood is taken as 6, which is decreasing based on the number of iterations.

6 Results and Discussion

Following section describes the results obtained for the problem domain using developed Single Architecture Single Processor and Single Architecture Multiple Processor Simulators.

6.1 Results from Single Architecture Single Processor Simulator

The developed simulator is executed with the last argument as 21, which means there will be 21 different trained network models. The order of input in each model is different. Final output will have: (a) The processor number and the corresponding output file created by that processor, (b) The name of the output file and corresponding percentage error, (c) The time taken by that processor to execute the job. This process is carried out for resubstitution, hold-out and K-fold evaluation methods. A comparison is made between the percentage error and the time taken by different network models and between different evaluation techniques.

Result for original data set of 2493 examples

The various parameters used in neural network model are given in Table 1 and comparative performance is tabulated in Table 2.

Table 1. Parameters of neural network

Parameter	Parameter Value
Number of examples	2493
Number of Input units	3
Number of Kohonen Units	12 x 12
Number of Output units	2 x 144
Number of Epochs	100
Learning Rate	.05
Width of Topological Neighborhood	6.0

Time taken by hold-out method is minimum, which is obvious, as the network is trained for only 67% of data and tested for only 33% on data. Time taken by K-fold method is higher than other methods because the data set is divided into 10 subsets, so training and testing is done 10 times by each processor. Average error is around 5.5%. Which means out of

2493 examples only around 135 examples are misclassified, which is reasonable when comparing to such a large amount of data set.

Table 3 shows the number of misclassified examples in the testing dataset, by each evaluation methods.

Table 2. Comparison of computational time and error for different evaluation methods

Sr. No.	Resubstitution		K - fold		Hold-out	
	Error (%)	Time (s)	Error (%)	Time (s)	Error (%)	Time (s)
1	4.89	172	5.01	1647	6.80	114
2	5.25	170	5.46	1526	4.37	117
3	5.17	168	5.98	1525	6.56	110
4	5.70	213	5.30	1582	4.13	135
5	5.66	210	5.01	1511	4.37	146
6	5.58	210	5.05	1401	4.62	136
7	6.30	217	5.30	1542	7.05	149
8	6.22	217	5.42	1511	5.71	146
9	4.49	214	5.02	1414	7.05	149
10	5.50	170	5.66	2066	5.10	115
11	5.90	168	4.73	1888	4.86	116
12	6.22	166	4.85	1925	5.47	120
13	4.89	161	5.66	2125	5.10	115
14	5.62	167	5.13	1882	4.86	117
15	6.38	165	5.14	1953	4.37	114
16	5.70	231	5.54	2190	7.05	156
17	5.05	215	5.26	1902	5.83	160
18	5.17	224	5.57	1962	5.10	143
19	6.30	213	5.54	2111	4.13	138
20	6.38	214	4.29	1848	6.68	150
21	5.17	214	5.54	1961	5.71	114
Avg. Error	5.60		5.26		5.47	
Std. Dev.	0.55		0.37		1.01	
Total Time		11min. 22 sec.		101 min. 5 sec.		7 min. 55 sec.

Table 3. Missclassified examples

Data Set: 2493 examples				
Evaluation Method (↓)	Training Set	Testing Set	Average Error (%)	Number of misclassified examples
Resubstitution	2493	2493	5.60	140
Hold-out	1670	823	5.47	45
K-fold	2243	250	5.26	14

Average error of resubstitution is found to be more than that of K-fold and hold-out. But as in resubstitution the network is tested for the same data set for which it is trained, so the error has to be less than the error from other methods, where network is tested for new data. This raised the question about the accuracy of the original data set. The features of some examples may have some abnormality. To check this, a frequency graph (Figure 1) between "Numbers of times a example is misclassified in 21 different architectures" Vs "Number of examples" is drawn.

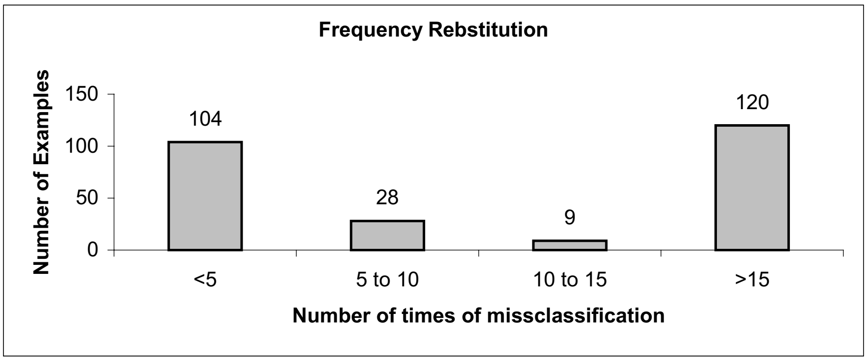


Fig. 1. Number of examples Vs number of times of misclassification

The graph shows that around 120 out of 2493 examples are misclassified more than 15 times in 21 runs (i.e. misclassified 15 times in 21 different architectures). Also there are 90 examples, which are misclassified for all 21 different architectures. So, we can say that some abnormality may exist in these examples. To verify this abnormality, data analysis is carried out.

Data Analysis

Data Analysis is the process of systematically applying statistical and logical techniques to describe, summarize, and compare data. It helps in understanding information (data). This process involves organizing and examining the collected data using narratives, charts, graphs or tables. Here 2D and 3D graphs are used (Agrawal 2004). Let **D** be the original data set of 2493 examples. Let 'M' be the data set of above 120 examples, which are misclassified more than 15 times. As the current weld data set has three features, so it is possible to represent the whole data in 3D. From the 2D and 3D plot of the data set it is seen that all the misclassified examples either lies in the boundary of the data set D or scattered away from the main cluster of data (Agrawal 2004). This explains why these examples are always misclassified. Features are the only available characteristic of the weld image, so these abnormalities cannot be explained by the value of the features only.

Irrespective of the network these 120 examples are misclassified always. In resubstitution method all these examples are part of testing data set. This explains why the error of resubstitution is more than the Hold-Out and K-fold. Similarly in hold-out 33% of data are used for testing. So the probability of these 120 examples to be part in testing data set is greater than that in K-fold, where testing is done for only 10% of data. Again the models are trained for a new data set 'C', such that {C} = {D} – {M} i.e. {C} has 2493 – 120 = 2373 examples. Following section highlights the result for {C} data set.

Results for data set of 2373 examples

The various parameters used in Neural Network model are same as given in Table 1. Only number of examples considered is changed from 2493 to 2373. The results are tabulated in Table 4.

The average error of resubstitution decreased from 5.60 to 1.70 i.e. the percent accuracy is 98.3%. So, out of 2373 examples only 41 examples are misclassified. As the size of data set decreased the time taken to train a network also decreased. Table 5 shows the number of misclassified examples in the testing data set, by each evaluation technique. So, the overall performance of every network is improved by removing only 120 examples out of 2493. This also explains the importance of the data analysis for model performance validation.

6.2 Single Architecture Multiple Processor

The various parameters used in Neural Network model are same as given in Table 1. Only number of examples considered is changed from 2493 to 2373. The simulator took around 13.2 minutes to train a network. While in Single Architecture Single Processor Simulator, the time required to train a network with the same parameters of network is around 200s. So, for the LVQ type of classification Single Architecture Single Processor simulator is far better than the Single Architecture Multiple Processor Simulator. In the LVQ algorithm, best matching, or winning neuron for an input vector is one having the minimum values among all the neurons.

Table 4. Comparison of computational time and error for different evaluation methods

Sr. No.	Resubstitution		K - fold		Hold-out	
	Error (%)	Time (s)	Error (%)	Time (s)	Error (%)	Time (s)
1	2.11	105	4.631	1121	4.85	72
2	1.6	105	7.165	976	3.83	71
3	1.6	105	5.06	965	3.19	71
4	1.98	164	3.961	947	3.7	71
5	1.18	16	4.843	956	4.08	109
6	1.73	115	2.905	1705	4.59	110
7	1.98	155	5.008	1470	6.12	108
8	2.82	159	4.087	1376	2.55	134
9	1.47	157	2.905	1696	4.85	145
10	1.73	163	3.667	1469	4.08	107
11	1.31	161	4.171	1308	1.66	113
12	1.73	157	6.278	1676	3.57	107
13	1.73	164	3.833	1512	8.04	109
14	1.31	166	4.802	1353	0.51	112
15	1.56	159	6.278	1649	3.44	107
16	1.98	183	3.368	1490	2.55	138
17	1.1	169	3.538	1349	0.77	139
18	1.81	105	4.168	2253	5.87	109
19	1.98	178	4.216	1883	8.04	109
20	1.85	167	4.168	2169	0.51	110
21	1.1	155	4.455	1892	4.97	104
Avg. Error	1.70		4.45		3.89	
Std. Dev.	0.39		1.06		2.06	
Total Time		8 min. 35 sec.		83 min. 2 sec.		6 min. 40 sec.

Table 5. Evaluation technique and misclassified examples

Data Set: 2373 examples				
Evaluation Method (↓)	Training Set	Testing Set	Average Error (%)	Number of misclassified examples
Resubstitution	2373	2373	1.70	41
Hold-out	1590	783	3.89	30
K-fold	2135	238	4.45	11

These neurons are distributed among all the available processors in Single Architecture Multiple Processor Simulator. All processors pass their local winner to Master Processor, which then calculates the global winner, and pass it to other processor. So, in each iteration, all processors have to communicate with each other. Hence, passing of information takes time. The time required in one iteration, by 8 processors to do the calculation jointly, is more than that required by a single processor to do the same calculation alone. The Single Architecture Single Processor concept can only be used when the time required to pass the information is less than the time required to do the calculation.

7 Summary

In the present paper, development of Linear Vector Quantization based supervised neural network classifier in parallel processing environment is discussed. Two variants of parallel neuro classifier were developed on PARAM 10000. They are Single Architecture Single Processor and Single Architecture Multiple Processor. Further, the weld defect classification application is demonstrated successfully using developed parallel classifiers. The study revealed interesting observations. Systematic data analysis helped in completing data set and also could eliminate the poor data examples. Clean data set gave better performance than original data set. Single Architecture Single Processor found to be better than Single Architecture Multiple Processor in terms of computational time.

Acknowledgments

Authors are grateful to Prof. T W Liao, Industrial and Manufacturing Systems Engineering Department, Louisiana State University, Baton Rouge, LA 70803, USA for providing weld defect dataset. Also, authors gratefully acknowledge the facility of PARAM 10000 installed by Center for Development of Advanced Computing (C-DAC) at IIT Kharagpur and provided to them for the reported research work.

References

Agrawal, P. (2004), *Development of Parallel Neuro Classifier for Weld Defect Classification Problem*, B.Tech Thesis, Department of Civil Engineering, IIT Kharagpur, India.

Barai S. V., and Reich, Y. (2002), "Weld classification in radiographic images: data mining approach", *Proceedings of National Seminar on Non Destructive Evaluation*, held at Chennai during 5-7 December 2002, and organized by Indian Society for Non Destructive Testing Chapters in Chennai, Kalpakkam and Sriharikota.

Berger, H. (1977), *Nondestructive Testing Standards - A Review*, STP 624, ASTM.

Bray, D. E. and Stanley, R. K. (1989), *Nondestructive evaluation - A tool for Design, Manufacturing and Service,* McGraw-Hill Book Company

Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P. and Uthursamy, R. (1996), *Advances in knowledge discovery and data mining*, AAAI Press/The MIT Press, Cambridge, MA.

Haykin, S. (1994), *Neural networks - A comprehensive Foundation*, Macmillan College Publishing Company, New York, USA.

Kohonen, T. (1995), *Self-Organizing Maps*, Berlin: Springer-Verlag.

Labonte, G. and Quintin, M. (1999), "Network Parallel Computing for SOM Neural Networks", (Royal Military College of Canada), http://www.rmc.ca/academic/math_cs/labonte/HPCS99.pdf

Liao, T. W. and Li, D. (1997), " Two manufacturing applications of the fuzzy k-NN algorithm", *Fuzzy Sets and Systems*, Vol. 92, pp: 289-303.

Liao, T. W., Li, D. M. and Li, Y. M. (1999), " Detection of welding flaws from radiographic images with fuzzy clustering methods", *Fuzzy Sets and Systems*, Vol. 108, pp: 145-158.

Liao, T. W. and Li, Y. (1998), " An automated radiographic NDT system for weld inspection: Part II – Flow detection", *NDT&E International,* Vol. 31, No. 3, pp: 183-192.

Liao, T. W. and Ni, J. (1996), " An automated radiographic NDT system for weld inspection: Part I – Weld Extraction", *NDT&E International*, Vol. 29, No. 3, pp: 157-162.

Liao, T. W. and Tang, K. (1997), " Automated extraction of welds from digitized radiographic images based on MLP neural networks", *Applied Artificial Intelligence*, Vol. 11, pp: 197-218.

Liao, T. W., Wang, G., Triantaphyllou, Chang, P. C. (2001), "A data mining study of weld quality models constructed with MLP neural networks from stratified samples data", Industrial Engineering Research Conference, Dallas, TX, May 20-23, 2001, http://cda4.imse.lsu.edu/PROCEEDINGS_Papers/IERC2001paper.PDF

Lippman, R. P. (1987), "An introduction to computing with neural nets", *IEEE ASSP Magazine*, 4, 6-22.

Nafaa, N. Redouane, D. and Amar, B. (2000), "Weld defect extraction and classification in radiographic testing based artificial neural networks", *15th WCNDT, Roma 2000*, http://www.ndt.net/article/wcndt00/papers/idn575/idn575.htm

Rao, B. P. C., Raj, B. and Kalyansundaram, P. (2002), "An artificial neural networks for eddy current testing of austenitic stainless steel welds", *NDT & E International*, Vol. 35, pp: 393-398

Reich, Y. (1997), Machine learning techniques for civil engineering problems, *Microcomputers in Civil Engineering*, 12, 4, 307-322.

Reich, Y. and Barai, S. V. (1999), Evaluating machine learning models for engineering problems, *Artificial Intelligence in Engineering*, 13, 257-272.

Reich, Y. and Barai, S. V. (2000), A methodology for building neural networks model from empirical engineering data, *Engineering Applications of Artificial Intelligence*, 13, 685-694.

Seiffert, U. (2002), "Artificial Neural Networks on Massively Parallel Computer Hardware", (University of Magdeburg, Germany). *Proceedings of European Symposium on Artificial Neural Networks Bruges* (Belgium), 24-26 April 2002.

Stepinski, T. and Lingvall, F. (2000), "Automatic defect characterization in ultrasonic NDT", *15th WCNDT*, *Roma 2000*, http://www.ndt.net/article/wcndt00/papers/idn393/idn393.htm

Appendix: Brief Introduction to PARAM 10000

PARAM 10000 is a 6.4 GF parallel computer, which is scalable up-to tera flop range, developed by Center for Development of Advanced Computing (C-DAC), India. The processors of PARAM 10000 belong to Sun Enterprise 250 family. Sun Enterprise 250 server accommodates two 400 MHz UltraSPARC II processors for extra-high performance. PARAM 10000 has three compute nodes and one server node. The total configuration consists of eight processors. The server is named e250a & the compute nodes are named as e250b, e250c & e250d. Each node in PARAM 10000 is a dual processor SMP system having 2MB of level-2 cache per CPU.

An Innovative Approach to Genetic Programming–based Clustering

I. De Falco[1], E. Tarantino[1], A. Della Cioppa[2] and F. Fontanella[3]

[1]Institute of High Performance Computing and Networking – CNR
Via P. Castellino, 111 80131 Naples – Italy ivanoe.defalco@na.icar.cnr.it
[2]Dept. of Computer Science and Electrical Engineering University of Salerno
Via Ponte don Melillo, 1 84084 Fisciano (SA) – Italy adellacioppa@unisa.it
[3]Dept. of Information Engineering and Systems
University of Naples, Via Claudio, 21 80125 Naples – Italy frfontan@unina.it

Summary. Most of the classical clustering algorithms are strongly dependent on, and sensitive to, parameters such as number of expected clusters and resolution level. To overcome this drawback, a Genetic Programming framework, capable of performing an automatic data clustering, is presented. Moreover, a novel way of representing clusters which provides intelligible information on patterns is introduced together with an innovative clustering process. The effectiveness of the implemented partitioning system is estimated on a medical domain by means of evaluation indices.

Key words: Genetic Programming, cluster representative, data clustering

1 Introduction

Clustering is the part of data mining whose task consists in grouping a set of similar objects based on some measures of goodness that differ according to application [1, 2]. Differently from the supervised learning in which, given a collection of labeled training examples used to learn the descriptions of classes, the problem is to label a newly encountered pattern, in clustering the problem is to group a given collection of unknown patterns into meaningful clusters with no assumptions about the relationships.

The clustering aim is to develop methods and tools for analyzing large data sets and for searching for unexpected relationships in the data. A variety of clustering algorithms have been developed to face the problem [3]. Most of these algorithms require user inputs of several parameters like the number of clusters and the average dimensionality of the cluster [4, 5, 6], which are not only difficult to determine but are also not practical for real–world data sets. In the hierarchical algorithms a human user evaluates a posteriori the resolution level, and thus the cluster number, by which he wants to partition the assigned data set. In other cases, different criteria have been introduced to find the optimal value of the cluster number [7, 8, 9]. Hence, the output of these clustering algorithms is very sensitive to user expertise.

To overcome these drawbacks, evolutionary techniques have been applied to cluster analysis [10, 11, 12, 13]. Moreover, the usage of combined strategies has been attempted [9, 14]. In this paper the effectiveness of a Genetic Programming (GP) [15] framework to perform automatic data clustering without providing any kind of user knowledge is investigated. The hope is that GP, based only on information implicitly contained in the data set, allows to find an efficient partitioning with interesting correlations among patterns. To this aim an innovative way of representing clusters, based on logical formulas, is introduced. Furthermore a new clustering procedure tied to this representation is proposed. A solution is found by maximizing intra–cluster homogeneity and inter–cluster separability of the clustering. This evolutionary system is applied to clustering in a medical domain.

The paper is organized as follows: in section 2 a formalization of data clustering is reported, while in section 3 our GP–based clustering system is outlined together with its implementation details. In section 4 the test problem and some evaluation indices are described. The performance of our system is discussed in section 5. The last section illustrates our final remarks and future work.

2 Data Clustering

Definitions and notations. Each element of the database is said *pattern* and is represented by $\mathbf{X} = (x_1, \dots, x_\ell)$ with $\mathbf{X} \in \mathbf{S}$, where $\mathbf{S}$ is the universe of all possible elements with ℓ attributes and x_j denotes the j–th attribute of the pattern. A *pattern set* $\mathcal{X}$ with cardinality n is denoted with $\mathcal{X} = (\mathbf{X}_1, \dots, \mathbf{X}_n)$ with $\mathcal{X} \subseteq \mathbf{S}$.

Distance between patterns. Definition of distance between patterns depends on the type of attributes. For binary attributes the Hamming distance is used while for numerical ones the linear distance $\delta(x_j, y_j) = |x_j - y_j|$ is considered.

Once defined the range for the j-th attribute as:

$$R_j \equiv \delta(\max_{i=1,\dots,n} x_{ij}, \min_{i=1,\dots,n} x_{ij})$$

where x_{ij} indicates the j–th attribute of the i–th pattern, the distance between patterns chosen is the Manhattan one:

$$d(\mathbf{X}, \mathbf{Y}) = \frac{1}{\ell} \times \sum_{j=1}^{\ell} \left(\frac{1}{R_j} \times \delta(x_j, y_j) \right) \tag{1}$$

This distance is normalized with respect to the range R_j and to the number of attributes, so as to have $0 \le d(\mathbf{X}, \mathbf{Y}) \le 1 \;\; \forall \;\; \mathbf{X}, \mathbf{Y} \in \mathcal{X}$.

Cluster representative and clustering. Several ways are possible to represent a cluster $\mathbf{C}_k$. In the enumerative method for each cluster the data set elements which belong to it are listed. In others a cluster is represented by listing for each attribute the maximum and the minimum values computed on cluster members. In many algorithms a representative for any cluster $\mathbf{C}_k$ is used, i.e. an element of set $\mathbf{S}$ which can whether or not belong to the cluster. However, the most used representative is the centroid $\mathbf{P}_k$ defined as the average point for patterns belonging to $\mathbf{C}_k$. In this case, a clustering $\mathbf{CL}$ with cardinality $\mathcal{N}$ can be denoted by the list of its cluster representatives: $\mathbf{CL} \equiv \{\mathbf{P}_1, \ldots, \mathbf{P}_{\mathcal{N}}\}$.

Scoring function, homogeneity and separability. To quantify the clustering quality with respect to data set taken into account, a scoring function f_s, expressed in terms of homogeneity $\mathcal{H}$ and separability $\mathcal{S}$ for the clustering $\mathbf{CL}$, is considered. Thus a generic score function is $f_s = f(\mathcal{H}(\mathbf{CL}), \mathcal{S}(\mathbf{CL}))$ where the dependence on $\mathcal{X}$ is implicit in the computation of $\mathcal{H}(\mathbf{CL})$ and $\mathcal{S}(\mathbf{CL})$. Denoting with w_k the cardinality for a cluster $\mathbf{C}_k$, homogeneity is defined as:

$$\mathcal{H}(\mathbf{C}_k) \equiv -\frac{\sum_{\mathbf{X}\in\mathbf{C}_k}[d(\mathbf{X}, \mathbf{P}_k)]}{w_k} \tag{2}$$

Hence we can define clustering homogeneity as weighted average of homogeneity of clusters, and separability for a clustering as the weighted average of distances among clusters. Formally, we have:

$$\mathcal{H}(\mathbf{CL}) \equiv \frac{\sum_{i=1}^{m} w_i \times \mathcal{H}(\mathbf{C}_i)}{\sum_{i=1}^{m} w_i}; \;\; \mathcal{S}(\mathbf{CL}) \equiv \frac{\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} w_i \times w_j \times d(\mathbf{C}_i, \mathbf{C}_j)}{\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} w_i \times w_j} \tag{3}$$

Distance between clusters is computed as the distance between respective centroids by using (1).

3 Our Genetic Programming System for Data Clustering

Clustering encoding. In our system, a cluster prototype is a logical formula E constituted by a variable number of predicates and a clustering is represented by a variable number of such formulas. Any GP individual encodes a clustering as a tree structure. A formula generator, based on the context–free grammar in Table 1, provides the starting population. This grammar ensures the syntactic correctness of the formulas. The tree nodes can be either terminal or nonterminal symbols. These latter are indicated in capital letter. Moreover, the terminal symbol \$ is introduced as formula delimiter. Since the grammar is non–deterministic, the action carried out by a rule is chosen based on the fixed values of probabilities shown in the table so as to reduce the probability of generating too long formulas. However, in addition an upper limit has been imposed on the total number of predicates contained in the set of formulas representing a clustering and individuals which overcome it will be generated anew. The generator starts by applying the starting rule S and then the remaining ones as long as they are called upon. The aim is to create a set of logical formulas different in terms of size, shape and functionality. It can be noted that S allows to generate individuals composed by at least two formulas. For the predicates of each formula E_k conditions on some of the attributes a_i of the database patterns are set. When all the predicates of this formula are satisfied by the attribute values of the pattern, the formula is said to match the pattern itself. The matching between a generic pattern and the formulas is effected by an interpreter. All the formulas of the individual will be used to perform the clustering process which takes place in two steps:

1. assign all the patterns which are matched by the formulas (the formulas which match no pattern are not considered in the step 2 and they simply represent introns in the genotype):
   - a pattern is matched by one formula only: it is assigned to the related cluster;
   - a pattern is matched by more than one formula with different number of predicates: the pattern is assigned to the formula with the greatest number of matched predicates[1];
   - a pattern is satisfied by more than one formula with the same number of predicates: the pattern is classified as undefined and it is not assigned at this stage;
2. assign the patterns which are not matched by any formula of the clustering and the undefined ones:
   - for each formula, compute the average points of all the patterns assigned so far to it: this point is called *matching centroid*;
   - assign any pattern to the cluster whose matching centroid is the closest.

This process, hereinafter referred to as *clustering*, permits to combine the advantages of a clustering whose representation is based on logical formulas with those of a classical representation based on centroids. In fact, the former provides a feature extraction and correlation capability and the latter makes the process able to perform a complete clustering in which all the patterns are assigned. Hence, based on what expressed above, the phenotype $\mathbf{F}$ is constituted by a variable number of formulas, each of which is representative of a cluster. Formally, the clustering is represented by $\mathbf{F} \equiv \{E_1, E_2, \ldots, E_{\mathcal{N}}\}$ where $\mathcal{N}$ is the number of formulas.

Fitness function. After performing the *clustering* process, an appropriate fitness function must be chosen to evaluate the quality of the segmentation based on $\mathbf{F}$. A penalty term linked to the number of clusters could be added in the fitness, with the aim to avoid a probable overfitting of the solution with respect to the given database due to uncontrolled increase in the number of

[1] The aim is to assign the pattern to the formula which delimits a smaller part of feature space, so to obtain a more specialized cluster.

Table 1. The grammar for the random program generator.

Rule number	Rule	Probability
1	$S \longrightarrow (Q)BA\$(Q)BAE$	1.0
2	$Q \longrightarrow L\|M$	equiprobable
3	$L \longrightarrow C \wedge L\|C$	0.7, 0.3
4	$M \longrightarrow C \vee M\|C$	0.7, 0.3
5	$B \longrightarrow \vee\|\wedge$	equiprobable
6	$A \longrightarrow (Q)BA\|(Q)$	0.3, 0.7
7	$E \longrightarrow \$(Q)BAE\|\$$	0.6, 0.4
8	$C \longrightarrow (PNZ)$	1.0
9	$P \longrightarrow a_0\|a_1\|\ldots\|a_{32}$	equiprobable
10	$Z \longrightarrow 0\|1\|2\|3$	equiprobable
11	$N \longrightarrow \geq \| \leq \| = \| > \| <$	equiprobable

clusters. Nevertheless, this would mean to establish a limit to this number, by using some a priori knowledge on the application domain, which would make our algorithm dependent on user skill. To surmount this problem, following [16], we consider as fitness function a linear combination of intra–cluster homogeneity and inter–cluster separability:

$$f(\mathbf{F}) = \mathcal{H}(\mathbf{F}) + \mu \times \mathcal{S}(\mathbf{F}) \tag{4}$$

where μ is a scaling factor. Since $d(\mathbf{X}, \mathbf{Y}) \leq 1$, we have that $-1 \leq \mathcal{H}(\mathbf{F}) \leq 0$ and $0 \leq \mathcal{S}(\mathbf{F}) \leq 1$. Therefore for the fitness it results that $-1 \leq f(\mathbf{F}) \leq \mu$. In this way, data clustering becomes a problem of direct maximization for homogeneity and separability independently of number of clusters $\mathcal{N}$, and depending on a scale factor liable for balance between them. Thus the fitness $f(\mathbf{F})$ does not explicitly depend on $\mathcal{N}$, yet as the parameter μ varies a control on the number of clusters will be indirectly achieved. In fact the relative weights for $\mathcal{H}$ and $\mathcal{S}$ change as μ varies. Namely it is to note that, for low values of μ, $\mathcal{H}$ is dominant in (4) and as $\mathcal{H}$ increases $\mathcal{N}$ tends to grow, while, when μ grows, the value of $\mathcal{S}$ assumes a more significant weight in (4) and, consequently, its increment tends to let $\mathcal{N}$ decrease.

To calculate more accurately $\mathcal{H}(\mathbf{F})$ and $\mathcal{S}(\mathbf{F})$, the centroids of all the actual clusters achieved at the end of the *clustering* process are evaluated and their values are used in (3).

Genetic operators. The phenotypes are encoded as derivation trees, generated by the grammar, representing the genotypes the genetic operators work with. This encoding is very appealing in that the actions performed by the genetic operators can be easily implemented as simple operations on the trees. The new elements in the population are generated by means of two operators, *crossover* and *mutation*, which preserve the syntactic correctness of the formulas. The crossover operates by randomly selecting both a nonterminal node in the first individual to be crossed and the same nonterminal node in the second individual. Then, it swaps the derivation subtrees. If a corresponding nonterminal node cannot be found in the second parent, the crossover takes place on different nodes. Differently from classical GP, the mutation works on any obtained offspring by randomly choosing a nonterminal node in the individual to be mutated, and then the corresponding production rule is activated in order to generate a new subtree. Depending on the nonterminal symbol chosen, this operation can result either in the substitution of the related subtree (macro–mutation) or in a simple substitution of a leaf node (micro–mutation).

4 Evaluation Indices and Database

Evaluation indices. Indices can be defined and exploited to quantitatively estimate a posteriori the degree of usefulness and meaningfulness of found clusters. In general, given a clustering and a data set where classes are known, a table between the p classes and the s found clusters can be conceived. This table corresponds to a matrix $\mathbf{B}$ where element b_{ij} represents the number of patterns in the data set belonging to class i assigned to cluster j. Two indicators can be defined: the Class Addressing Index $\mathcal{I}$ and the Class Gathering Index $\mathcal{G}$. For the j-th cluster:

$$\mathcal{I}_j = \sum_{i=1}^{p} \frac{b_{ij}^2}{(\sum_{i=1}^{p} b_{ij})^2} \tag{5}$$

This index represents a normalized weighted average and it measures the ability of a cluster to address towards a single class. Its value is 1 in the best case, in which the cluster represents only one class, and it decreases as the number of the addressed classes increases. In the worst case its value is equal to $\frac{1}{p}$. We define $\mathcal{I}$ for the whole clustering as the weighted average for indices of each cluster:

$$\mathcal{I} = \frac{\sum_{j=1}^{s} w_j \times \mathcal{I}_j}{\sum_{j=1}^{s} w_j} \tag{6}$$

where w_j denotes the weight of j–th cluster, i.e. the number of patterns assigned to j–th cluster. Its variation range is the same of $\mathcal{I}_j$.

In a similar way, for the i–th class $\mathcal{G}$ is:

$$\mathcal{G}_i = \sum_{j=1}^{s} \frac{b_{ij}^2}{(\sum_{j=1}^{s} b_{ij})^2} \tag{7}$$

It indicates as a single class is subdivided among the clusters. Also in this case, its value is 1 when all the elements of one class are grouped in one cluster, while it decreases down to $\frac{1}{s}$ as the number of clusters in which a class is subdivided increases. For all the clustering

$$\mathcal{G} = \frac{\sum_{i=1}^{p} w_i \times \mathcal{G}_i}{\sum_{i=1}^{p} w_i} \tag{8}$$

Table 2. Average values of homogeneity and separability as a function of μ

μ	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
$\mathcal{H}$	-0.106	**-0.105**	-0.111	-0.115	-0.119	-0.120	-0.119	-0.125	-0.136	-0.140	-0.133
$\mathcal{S}$	0.234	0.237	0.244	0.248	0.253	0.254	0.252	0.259	0.265	**0.274**	0.265
$\mathcal{N}$	15.6	14.6	12.2	9.8	7.5	6.7	4.9	4.5	4.3	4.2	3.4

where w_i denotes the weight of i–th class, i.e. the number of patterns belonging to i–th class. The variation range is the same as that of $\mathcal{G}_i$.

For a global evaluation we take into account the so-called Correspondence Index $\mathcal{I}_c = \mathcal{I} \times \mathcal{G}$. Its range is $(1/p) \times (1/s) \leq \mathcal{I}_c \leq 1.0$.

The database. A database, constituted by 366 clinical dermatological cases, available at UCI site [17] has been considered. Each instance has 34 attributes, of which 12 clinical and 22 hysto–pathological. Any attribute, apart from age and family anamnesis, is in the range $[0, 3]$ where 0 indicates the absence of the attribute, 3 indicates the presence at its maximum degree, while 1 and 2 denote intermediate values. Family anamnesis is a boolean attribute with value 1 if any dermatological pathology has been observed in the family, 0 otherwise. Such a database is already subdivided into 6 classes on the base of medical belief.

5 Experimental Findings

Preliminary trials have allowed to set the basic evolutionary parameters. In particular population size p_s=100, tournament selection with tournament size $\mathcal{T} = 10$, mutation probability p_m=0.9, crossover probability $p_c = 0.9$ and number of generations $n_g = 250$. Besides the maximum number of predicates in the set of formulas has been set equal to 200.

The experiments aim at studying the variation of $\mathcal{H}$ and $\mathcal{S}$ as a function of the scale factor μ in the range $[0.0, 1.0]$ with step 0.1. In Table 2 the average values of $\mathcal{H}$, $\mathcal{S}$ and $\mathcal{N}$ for each value of μ are reported. These values have been obtained performing 10 runs for each value of the scale factor. The objective is to find the best μ in terms of $\mathcal{H}$ and $\mathcal{S}$. In order to individuate it we have defined a theoretical 'optimal point' (shown in bold in table) obtained considering the best of homogeneity and separability among those found during our trials. Among all the different values of $\mathcal{H}$ and $\mathcal{S}$ in table, the closest to this 'optimal point', in terms of distance, has resulted to be that with $\mu = 0.7$. Table 2 shows also that as μ increases, the average number of clusters decreases which reveals that an implicit control on $\mathcal{N}$ can be actually achieved by suitably setting values for the scaling factor.

By using $\mu = 0.7$, the behavior during the best run in terms of fitness is shown in Fig. 1. In particular, on the left the average fitness of the population and the fitness of the best individual are reported as a function of the

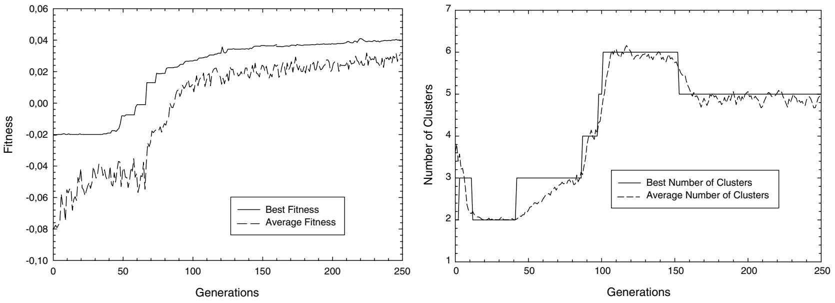


Fig. 1. Behavior of fitness (left) and number of clusters (right) during the best run

generation number. As it can be noted, there is a startup phase for the best fitness in which its value is almost constant and much higher than the average one. Such a behavior is due to the predominance of the separability in the fitness function which keeps low the related best number of clusters as it can be seen in Fig. 1 (right). Starting from about 40 generations this fitness increases constantly until generation 100. This is owed to the improvements of the homogeneity values as confirmed by the increase in the corresponding number of clusters with respect to the initial phase. Then the process continues with a lower increase until about 150 generations. During this phase the number of clusters remains constant. The last phase, up to generation 250, shows no significant variation in terms of the best fitness and the decrease and the stabilization in the associated number of clusters. In Table 3 the clustering obtained during the best run above is reported. For the medical database considered, it should be observed that the second and the fourth cluster address perfectly the first and the third class respectively. Almost the same holds for the first and the third cluster. Moreover it is clear that the second and the fourth classes have not been distinguished. Nonetheless, this result was expected since it is known that these classes are hard to separate by medical experts as well. This fact yields that the value of $\mathcal{I} = 0.8367$ is satisfactory but not so close to the maximum. Furthermore, it is worth observing that each class has been almost optimally gathered in one cluster. This is confirmed by the very good value of $\mathcal{G} = 0.989$. Thus the global index $\mathcal{I}_c$ is equal to 0.8275. These values assure that the clustering achieved is effective. As an example of the clustering, we report in the following only the formulas representative of the first two clusters in Table 3:

Cluster 1: $((a_{17} = 2) \vee (a_{10} = 1) \vee (a_{15} > 0)) \wedge ((a_{27} = 1) \vee (a_6 > 1)) \wedge ((a_{16} = 2) \vee (a_2 = 2) \vee (a_{30} = 3) \vee (a_{19} = 1) \vee (a_3 = 2) \vee (a_{15} < 2)) \wedge (a_{14} \geq 2)$
Cluster 2: $(a_{21} > 1) \wedge ((a_{22} > 1) \vee (a_{26} > 1) \wedge (a_0 > 1) \vee (a_{28} > 0) \vee (a_{21} < 3) \vee (a_0 < 3) \vee (a_{12} = 1) \vee (a_{18} > 1) \vee (a_{31} \leq 2) \vee (a_0 < 2) \vee (a_{11} > 1) \vee (a_{24} > 1)) \wedge ((a_{29} = 0) \vee (a_{18} < 2) \vee (a_{10} < 3) \vee (a_{12} < 2))$

Table 3. Classes–clusters ($p\backslash s$) relationship for the solution with $\mu = 0.7$

$p\backslash s$	1	2	3	4	5
1	0	112	0	0	0
2	0	0	0	0	61
3	0	0	0	72	0
4	0	0	0	0	49
5	50	0	0	0	2
6	0	0	19	0	1

Table 4. Average index values as a function of μ

μ	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
$\mathcal{I}$	0.5945	0.925	0.803	0.927	0.746	0.8154	0.8274	0.8167	0.665	0.6642	0.615
$\mathcal{G}$	0.835	0.735	0.783	0.816	0.704	0.938	0.952	0.979	0.949	0.916	0.933
$\mathcal{I}_c$	0.4964	0.6799	0.6287	0.7564	0.5252	0.7648	0.7877	0.7995	0.6311	0.6084	0.5738

As it is evident this novel representation has the advantage of providing explicit information both on the features of the patterns satisfying each formula and on their values.

Let us use $\mathcal{I}_c$ index to evaluate the found solutions from a clustering standpoint. It permits to estimate the utility of detected clusters in addressing towards the classes. In Table 4 the average values of all the indices, related to the best solutions in terms of fitness during 10 runs, are reported as a function of μ. It should be observed that the value $\mathcal{I}_c$ obtained for $\mu = 0.7$ is always the best. This seems to validate a posteriori its choice.

6 Conclusions and Future Work

In this paper, the flexibility of a Genetic Programming framework in performing automatic clustering has been exploited. The implemented system is capable of detecting clusters orienting towards classes. As a data clustering algorithm, satisfactory results have been achieved. Furthermore, thanks to GP, an innovative mode of representing clusters, based on logical formulas, has been introduced. This has not only allowed to perform an original clustering process which exploits the advantages of both a new representation and a classical one, but it has also permitted the discovery of common relationships of patterns belonging to a same cluster.

The system will be tested on different databases to further ascertain its effectiveness. A comparison with other clustering algorithms in terms of the evaluation indices proposed will be carried on. Besides, it could be parallelized with the scope of achieving an improvement both in performance and in solution quality.

References

1. Fayyad U M, Piatetsky-Shapiro G, Smith P (1996) From data mining to knowledge discovery: an overview. In: Fayyad U M et al. (eds) Advances in knowledge discovery and data mining. AAAI/MIT Press, 1–34
2. Hand D J, Mannila H, Smyth P (1988) Principles of data mining. MIT Press
3. Han J, Kamber M (2001) Data mining: concepts and techniques. Morgan Kaufmann
4. Zhang T, Ramakrishnan R, Livny M. (1996) BIRCH: an efficient data clustering method for very large databases. Proceedings of the ACM SIGMOD Int. Conf. on Management of Data, 103–114
5. Guha S, Rastogi R, Shim K (1998) CURE: an efficient clustering algorithm for large databases. Proceedings of the ACM SIGMOD Int. Conf. on Management of Data, 73–84
6. Aggarwal C, Yu P S (2000) Finding generalized projected clusters in high dimensional spaces. Proceedings of the ACM SIGMOD Int. Conf. on Management of Data, 70–81
7. Bock H H (1996) Probability models in partitional cluster analysis. In: Ferligoj A, Kramberger A (eds) Developments in data analysis, 3–25
8. Fraley C, Raftery A (1998) How many clusters? Which clustering method? Answers via model–based cluster analysis. The Computer Journal 41(8):578–588
9. Lee C Y, Antonsson E K (2000) Dynamic partitional clustering using evolutionary strategies. Proceedings of the 3rd Asia–Pacific Conference on Simulated Evolution and Learning. IEEE Press, Nagoya, Japan
10. Jain A K, Murty M N, Flynn P J (1999) Data clustering: a review. ACM Computing Surveys 31(3):264–323
11. Hall L O, Ozyurt B, Bezdek J C (1999) Clustering with a genetically optimized approach. IEEE Trans. on Evolutionary Computation 3(2):103–112
12. Sarafis I, Zalzala A M S, Trinder P W (2002) A genetic rule–based data clustering toolkit. Proceedings of the IEEE Congress on Evolutionary Computation, 1238–1243. Honolulu, Hawaii, USA
13. Cristofor D, Simovici D A (2002) An information–theoretical approach to clustering categorical databases using genetic algorithms. Proceedings of the Second SIAM International Conference on Data Mining, 37–46. Washington
14. Babu G P, Marty M N (1994) Clustering with evolutionary strategies Pattern Recognition 27(2):321–329
15. Koza J R (1992) Genetic Programming: on programming computers by means of natural selection and genetics. The MIT Press, Cambridge, MA
16. Yip A M (2002) A scale dependent data clustering model by direct maximization of homogeneity and separation. Proceedings of the Mathematical challenges in scientific data mining IPAM, 14-18 January, www.ipam.ucla.edu/publications/sdm2002/sdm2002_ayip.pdf
17. Murphy P M, Aha D W UCI Repository of machine learning databases. University of California, Department of Information and Computer Science, www.ics.uci.edu/~mlearn/MLRepository.html

An Adaptive Fuzzy Min-Max Conflict-Resolving Classifier

Shing Chiang Tan[1] **M.V.C. Rao**[2] **Chee Peng Lim**[3]
Faculty of Information Science and Technology[1]
Faculty of Engineering and Technology[2]
Multimedia University
Melaka Campus, Jalan Ayer Keroh Lama, Bukit Beruang, 75450 Melaka
Malaysia
sctan@mmu.edu.my machavaram.venkata@mmu.edu.my

School of Electrical and Electronic Engineering[3]
University of Science Malaysia
Engineering Campus, 14300 Nibong Tebal, Penang
Malaysia
cplim@usm.my

Abstract. This paper describes a novel adaptive network, which agglomerates a procedure based on the fuzzy min-max clustering method, a supervised ART (Adaptive Resonance Theory) neural network, and a constructive conflict-resolving algorithm, for pattern classification. The proposed classifier is a fusion of the ordering algorithm, Fuzzy ARTMAP (FAM) and the Dynamic Decay Adjustment (DDA) algorithm. The network, called Ordered FAMDDA, inherits the benefits of the trio, *viz.* an ability to identify a fixed order of training pattern presentation for good generalisation; stable and incrementally learning architecture; and dynamic width adjustment of the weights of hidden nodes of conflicting classes. Classification performance of the Ordered FAMDDA is assessed using two benchmark datasets. The performances are analysed and compared with those from FAM and Ordered FAM. The results indicate that the Ordered FAMDDA classifier performs at least as good as the mentioned networks. The proposed Ordered FAMDDA network is then applied to a condition monitoring problem in a power generation station. The process under scrutiny is the Circulating Water (CW) system, with prime attention to condition monitoring of the heat transfer efficiency of the condensers. The results and their implications are analysed and discussed.

Keywords: Adaptive Resonance Theory, Ordering Algorithm, Fuzzy ARTMAP, Dynamic Decay Adjustment, Circulating Water System

1 Introduction

One of the motivations of combining artificial neural networks with other soft-computing techniques and perhaps the most fundamental intention is to boost the performance of the classifier by executing a novel mapping procedure towards a given dataset. According to Simpson (1992), a successful classifier should not only give good classification performance but also should comprise the following properties, i.e., to 1) learn a given task quickly; 2) overcome catastrophic forgetting; 3) solve nonlinearity separable problems; 4) provide the capability for soft and hard decisions regarding the degree of membership of the data within each class; 5) offer explanations of how the data are classified, and justification on why the data are classified as such; 6) perform generalisation that is independent of parameter tuning; 7) operate without knowledge of the distribution of the data in each class; and 8) settle conflicts resulting from overlaps of input space of different classes.

One of the neural network classifiers that possesses most of the properties above is Fuzzy ARTMAP (FAM) (Carpenter et al. 1992). The FAM network, which is a supervised model from the Adaptive Resonance Theory (ART) family, is capable of learning new data incrementally without forgetting previously learned knowledge. The FAM network can be trained in a unique fast learning mode. However, like other classifiers, the performance of FAM relies excessively on the presentation order of training data. To circumvent this problem, one approach is to feed random orders of training data presentation to the network, until a well-trained network has been obtained. Nevertheless, it is not an easy task to get a desired, well-trained network using this approach. In view of this, there is an ancillary procedure, called *ordering algorithm* (Dagher et al. 1999), to co-work with FAM to deal with this limitation. The ordering algorithm is based on the Max-Min clustering algorithm (Tou and Gonzalez 1976) and was proposed to process the training data by identifying a set of fixed order of training data presentation to the classifier. In Dagher et al. (1999), FAM was trained with a fixed order of input pattern presentation using the Max-Min clustering approach. For simplicity, the network is called Ordered FAM.

FAM learning incurs self-organising similar input patterns by means of hyper-rectangle categorical prototypes. The hyper-rectangular boxes are expanded for accommodating a new example of the training data. On the other hand, real-world datasets, which comprise instances of different class labels, occasionally have a certain degree of overlapping in the attribute space. Hence, it is conceivable that the presentation of these similar data samples but of different class labels to the network would be recognised as a set of distinctive yet conflicting prototypes. If such conflicting prototypes remain unsettled without the assistance of a categorical width contraction facility, it could result in undesirable effects to the generalisation capability of the network.

In this paper, a novel conflict-resolving adaptive network, which integrates the ordering algorithm, FAM, and the Dynamic Decay Adjustment (DDA) (Berthold and Diamond 1995) algorithm, is proposed. The fusion of these three techniques, known as Ordered FAMDDA, inherits the benefits of its predecessors, i.e., decreasing computational overhead by subscribing to one-pass training with a fixed order of input data presentation; fast, stable, and incrementally learning; and an ability to deal with overlapping prototypes of conflicting classes.

The proposed Ordered FAMDDA is evaluated using two benchmark problems and a real-world condition monitoring task. In particular, Ordered FAMDDA is used as an intelligent condition monitoring and faulty diagnosis tool to identify and to distinguish between faulty and normal operating conditions in a power generation station. The system under scrutiny in the power station is the Circulating Water (CW) system (System Description and Operating Procedures 1999). In this study we monitor the operating conditions of the condensers as the cooling water flowing through the system, with central attention focused on heat transfer problems, i.e., either good heat transfer or poor heat transfer, in the condensers. The organisation of this paper is as follows. In section 2, the ordering algorithm, FAM, DDA, and Ordered FAMDDA are briefly described. The operation of Ordered FAMDDA is described in section 3. The effectiveness of Ordered FAMDDA is demonstrated using two benchmark problems in section 4 and the results are compared with those of FAM and Ordered FAM. In section 5, the proposed Ordered FAMDDA network is employed to handle a condition monitoring problem of the CW system in a power generation station. Summary of the work is drawn in section 6.

2 The Ordering Algorithm, Fuzzy ARTMAP, and Dynamic Decay Adjustment Algorithm

Owing to limited space, the full description about the ordering algorithm, FAM and DDA will not be covered in detail. Interested readers can refer to Dagher et al. (1999), Carpenter and Grossberg (1987), Carpenter et al. (1991), Carpenter et al. (1992), and Berthold and Diamond (1995).

2.1 The Ordering Algorithm

The ordering algorithm is used to find the order of input data presentation for FAM learning. It is a type of Max-Min clustering algorithm with a pre-defined parameter setting in terms of the number of distinct classes of a classification task (i.e., n_{clust}). The ordering algorithm comprises three stages, which can be described as follows; In Stage 1, the complemented of the $M-$ dimensional $\boldsymbol{a}$ input pattern (Carpenter et al. 1992)

$$\boldsymbol{A} = \left(a_1, \cdots, a_M, a_{M+1}, \cdots, a_{2M}\right) \tag{1}$$

that maximises the sum

$$\sum_{i=1}^{M} \left| A_{M+i} - A_i \right| \tag{2}$$

is selected as the first pattern to be presented. This pattern is also treated as the first cluster centre of the training patterns. In Stage 2, the next $(n_{clust} - 1)$ input patterns are identified for presentation during the training phase. These patterns represent the next cluster centres of the training patterns. In this stage, the Euclidean distance between k remaining patterns and the existing cluster centres $\boldsymbol{A}^j$ are computed. The minimum Euclidean distance between the pattern and the cluster centre is then identified, as follows

$$d_{\min}^{A} = \min_{1 \le j \le k} \left\{ dist(\boldsymbol{A}, \boldsymbol{A}^j \right\} \tag{3}$$

The pattern, which has the maximum value of these distances, is selected as the next cluster centre. In Stage 3, the presentation orders for the remaining patterns are determined by finding the minimum Euclidean distances between these patterns and the n_{clust} cluster centres. The whole procedure of Stage 3 is repeated until all orders of the input pattern presentation for the network training phase have been identified.

2.2 Fuzzy ARTMAP (FAM)

FAM is an architecture composed of two Fuzzy ART (Carpenter et al. 1991) modules, ART$_a$ and ART$_b$, which are interconnected via a map field (Figure 1). Each ART module comprises three layers of nodes: $F_0^a\,(F_0^b)$ is the normalisation layer in which an $M-$dimensional input vector, $\boldsymbol{a}$, is complement-coded to a $2M-$dimensional vector, as in equation (1); $F_1^a\,(F_1^b)$ is the input layer which receives the complement-coded input vectors; $F_2^a\,(F_2^b)$ is the recognition layer which is a dynamic layer that encodes prototypes of input patterns and allows the creation of new nodes when necessary.

At ART$_a$, input $\boldsymbol{A}$ is propagated through F_1^a to F_2^a. Each neuron j in F_2^a is activated according to a choice function (Carpenter et al. 1992)

$$T_j = \frac{\left| \boldsymbol{A} \wedge \boldsymbol{w}_j^a \right|}{\alpha + \left| \boldsymbol{w}_j^a \right|} \tag{4}$$

where $\alpha \to 0$ is the choice parameter; $\boldsymbol{w}_j^a$ the weight of node j. Using a *winner-take-all* competition scheme, the neuron J with the largest activation is selected as the winning node. The key feature of FAM is the vigilance test which measures the similarity between the winning prototype patterns, $\boldsymbol{w}_J^a$, and $\boldsymbol{A}$ against a threshold (vigilance parameter, ρ_a) (Carpenter et al. 1992), i.e.

$$\frac{\left| \boldsymbol{A} \wedge \boldsymbol{w}_J^a \right|}{\left| \boldsymbol{A} \right|} \ge \rho_a \tag{5}$$

When the match criterion is satisfied, learning ensues (Carpenter et al. 1992)

$$\boldsymbol{w}_J^{a(new)} = \beta\left(\boldsymbol{A} \wedge \boldsymbol{w}_J^{a(old)}\right) + (1-\beta)\boldsymbol{w}_J^{a(old)} \tag{6}$$

where $\beta \in [0,1]$ is the learning rate. When the match criterion is not satisfied, a new node is created, with the input coded as its prototype pattern. As a result, the number of nodes grows with time, subject to the novelty criterion, in an attempt to learn a good network configuration autonomously. As different tasks demand different network structures, this learning approach thus avoids the need to specify a pre-defined static network size, or to re-train the network with a batch of new and old input patterns.

During supervised learning, ART$_a$ and ART$_b$, respectively, receive a stream of input and target patterns. FAM does not directly associate input patterns at ART$_a$ to target patterns at ART$_b$. Rather, input patterns are first classified into categorical prototypes before being linked with their target outputs via a map field. At each input pattern presentation, this map field establishes a link from the winning prototype in F_2^a to the target output in F_2^b. This association is used, during testing, to recall a prediction when an input pattern is presented to ART$_a$.

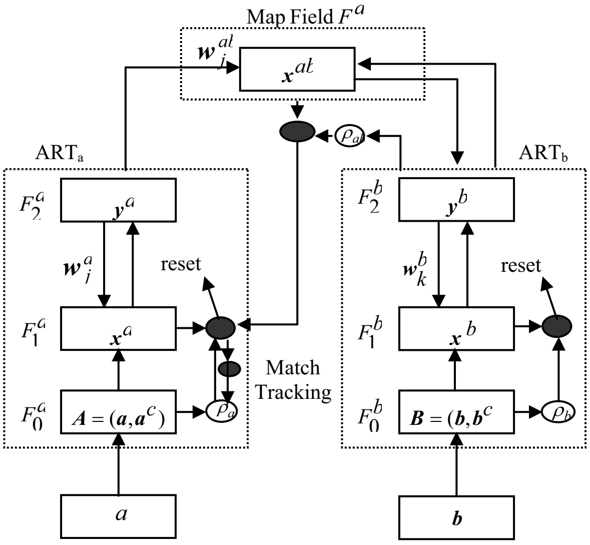


Figure 1. The architecture of FAM.

2.3 Dynamic Decay Adjustment (DDA) Algorithm

The DDA algorithm, which resorts to the constructive nature of the P-RCE algorithm (Riedmiller and Braun 1993) in providing a growing structure for the Radial Basis Function (RBF) (Moody and Darken 1989) network, is endowed with a capability of adjusting the width of the RBF prototypes locally. Width adjustment of the prototype is class dependent, which distinguishes the prototype from different neighbours. In general, the DDA algorithm comprises the following three steps:

- **Covered**. If a new pattern is correctly classified by an already existing prototype, it initiates regional expansion of the winning prototype in the attribute space.
- **Commit**. If a node of the correct class does not cover a new pattern, a new hidden node will be introduced, and the new pattern is coded as the reference vector.
- **Shrink**. If a new pattern is incorrectly classified by an already existing prototype of conflicting classes, the width of the prototype will be reduced (i.e. shrunk) for the sake of overcoming the conflicts.

3 The Ordered FAMDDA

There are several similarities between FAM and the DDA (Steps *Covered* and *Commit*) which motivate the fusion of the DDA into the FAM framework. First, as in Step *Covered*, learning for both FAM and the DDA incurs region expansion. In FAM, the learning process is deemed as the expansion of the winning node (hyper-rectangle) towards the pattern. Second, as in Step *Commit*, both FAM and the DDA perform incremental learning whereby new nodes are introduced when necessary. A new node will be committed to include the new input pattern if none of the existing nodes could classify it correctly. Furthermore, when FAM is in the fast learning mode, the weights of the newly committed node will be the same as the new input pattern.

In the proposed Ordered FAMDDA model, a number of modifications are necessary in order to implement the shrinking procedure effectively. A centre estimation procedure (Lim and Harrison 1997), is used whereby a new set of centre weight vector, $\boldsymbol{w}_j^{a-c}$, referred to as the *reference vector*, is introduced to each prototype in F_2^a. Each centre weight vector is initialised as

$$\boldsymbol{w}_j^{a-c} \equiv \left(w_{j1}^{a-c}, \cdots, w_{j2M_a}^{a-c}\right) = (0.0, \cdots, 0.0) \tag{7}$$

When learning takes place, the weight vector of the *J*-th F_2^a winning node is updated according to

$$\left(\boldsymbol{w}_J^{a-c}\right)^{new} = \left(\boldsymbol{w}_J^{a-c}\right)^{old} + \frac{1}{\left|\boldsymbol{w}_J^{ab}\right|}\left(\boldsymbol{A} - \left(\boldsymbol{w}_J^{a-c}\right)^{old}\right) \tag{8}$$

and

$$\left|\boldsymbol{w}_J^{ab}\right| = \left|\boldsymbol{w}_J^{ab}\right|^{\text{old}} + 1 \tag{9}$$

The procedure of the proposed Ordered FAMDDA network is as follows. First, the orders of presentation of the training data are determined using the ordering algorithm. On presentation of a training pattern to Ordered FAMDDA, the network will find a winning prototype, $\boldsymbol{w}_J^a$. The weights as well as the reference vector of prototype J are updated according to equations (6), and (8), and (9), respectively. After the FAM learning phase, a shrinking procedure ensues. Note that the shrinking procedure is applied successively to the current weights against the weights of the respective dimension of conflicting prototypes. If the current point, which represents the current weights of the $M-$dimension (or $2M-$dimension with complement coding) of the attribute space falls in the region formed by the prototype of conflicting classes, the width of the conflicting prototypical region is shrunk.

Three cases of width shrinking are proposed to overcome the conflict, and they are based on a heuristic to maximise the volume of each prototype. First, if the dimensions of the conflicting point can be shrunk without falling below a preset minimum width, $\varepsilon_{\min}$, the one with the smallest loss in volume $(\Lambda_{best,k})$ will be chosen, as follows:

$$\Lambda_{best,k} = \min\left\{\left|w_k^{a-c} - w_k^a\right|\right\} : \forall 1 \le j \le n, j \ne k :$$
$$\left(\frac{\Lambda_k - \left|w_k^{a-c} - w_k^a\right|}{\Lambda_k} \le \frac{\Lambda_j - \left|w_j^{a-c} - w_j^a\right|}{\Lambda_j}\right) \wedge \Lambda_k \ge \varepsilon_{\min} \tag{10}$$

Note that for $[0,1]-$normalised attributes, a choice of $1/10th$ is appropriate and this setting of $\varepsilon_{\min}$ stipulates that the parameter is uncritical (Huber and Berthold 1995). Second, if the above is not the case, either one of the remaining dimensions will be shrunk $(\Lambda_{\max,l})$, i.e.,

$$\Lambda_{\max,l} = \max\left\{\left|w_l^{a-c} - w_l^a\right|\right\} \tag{11}$$

Or, third, if the procedure results in a new width smaller than $\varepsilon_{\min}$, one of dimensions $(\Lambda_{\min,m})$ will be shrunk below $\varepsilon_{\min}$,

$$\Lambda_{\min,m} = \min\left\{\left|w_m^{a-c} - w_m^a\right|\right\} : \forall 1 \le j \le n, j \ne m :$$
$$\left(\frac{\Lambda_m - \left|w_m^{a-c} - w_m^a\right|}{\Lambda_m} \le \frac{\Lambda_j - \left|w_j^{a-c} - w_j^a\right|}{\Lambda_j}\right) \tag{12}$$

The conflict-free associations learned by Ordered FAMDDA are used, during the testing phase, to recall a prediction when an unseen pattern is presented to ART$_a$.

4 Benchmark Datasets: Experiments and Results

In this section, the classification performance of the Ordered FAMDDA network is evaluated using two benchmark databases from the UCI machine-learning repository (Blake and Merz 1998). These benchmark databases are Breast (Mangasarian and Wolberg 1990) and Glass (Murphy and Ana 1994), which are two-class and six-class classification problems, respectively. The experiments followed the procedure described in Dagher et al. (1999). Each dataset was divided randomly into a training set (2/3 of the data) and a test set (1/3 of the data). Noticeably, the percentage of data from each class in the training set was in accordance with the percentage of each class in the entire dataset. The number of the data used in the training set for Breast and Glass were 467 and 145, respectively. The number of the data used in the test set for Breast and Glass were 232 and 69, respectively. The order of training data presentation was processed with a setting of n_{clust} equalled to one more than the number of classes in the dataset. The parameters of Ordered FAMDDA were set at their "default" values, i.e., fast learning rate, $\beta = 1$; minimum width, $\varepsilon_{\min} = 0.1$; and ART$_a$ baseline vigilance, $\overline{\rho}_a = 0.0$. The intent of these parameter settings is in coherence with the design of Ordered FAMDDA that is independent of the tuning of parameters.

In each classification task, a total number of ten independent experiments were conducted using ten different ordered training sets and their corresponding test sets. In each experiment, the generalisation ability of Ordered FAMDDA in terms of test accuracy rate as well as the network size in terms of the number of nodes, were determined. To ascertain the stability of the performance, the upper and lower limits of average test accuracy rates were estimated at 95% confidence intervals using the bootstrap method (Efron 1979), which requires little modelling and assumptions compared to other standard methods.

The performances of Ordered FAMDDA for both Breast and Glass classification tasks are summarised in Table 1. The bootstrapped mean of test accuracy rates of Ordered FAMDDA were 96.90% and 73.17%, with intervals ranged from 96.46 to 97.37% and from 72.17 to 74.06%, respectively for Breast and Glass datasets. In other words, we are confident, at the 95% confidence level, that the mean of test accuracy rates fall within the intervals from 96.46% to 97.37%, and from 72.17% to 74.06%, respectively. These intervals are narrow, indicating that the network performance was stable.

Table 2 shows the performance comparison among FAM, Ordered FAM, and Ordered FAMDDA. Note that the FAM and Ordered FAM results are those published in Dagher et al. (1999). Several performance measures such as the average generalisation performance, the worst generalisation performance, the best generalisation performance, and the standard deviation of the generalisation performance were used to access the results of FAM, which was trained with ten different random orders of pattern presentation in each classification task. For FAM simulations, Dagher et al. (1999) repeated the experiments for three different collections of training/test data sets, apart from the first experiment whose results are presented in Table 2. The authors pointed that even though there were small discrepancies in numerical results using these three training/test data sets with the ones shown in Table 2, the conclusions obtained from these FAM results were of the same nature as the conclusions observed from Table 2. In this work, the bootstrapped mean of test accuracy rates of Breast and Glass classifications (from Table 1) are compared with the test accuracy rates of FAM and Ordered FAM. The performance of Ordered FAMDDA in Breast classification (96.90%) is better than those of FAM (96.12% for the best generalisation performance and 94.35%

for average generalisation performance) and Ordered FAM's (94.39%). Meanwhile, the performance of Ordered FAMDDA in Glass classification (73.17%) is better than that of Ordered FAM (69.56%) and is comparable to that of FAM (57.97–76.81%).

Table 3 depicts the network sizes that Ordered FAMDDA created. It is observed that the bootstrapped mean of the number of the nodes of Ordered FAMDDA are 7.3 and 25.7 with confidence intervals ranging from 6.9 to 7.7 and from 23.4 to 28.2, respectively, in Breast and Glass classifications. Also, notice that the mean network sizes for Ordered FAMDDA are lower than those of FAM and Ordered FAM in both classification tasks.

Table 1. Bootstrapped generalisation performance of Ordered FAMDDA.

Dataset	**Bootstrapped Accuracy (%)**		
	Lower Limit	**Mean**	**Upper Limit**
Breast	96.46	96.90	97.37
Glass	72.17	73.17	74.06

Table 2. Comparison of results from FAM (Worst Gen. – worst generalisation; Best Gen. – best generalisation; Avg. Gen. – average generalisation; std. dev. – standard deviation; Gen. – generalisation), and Ordered FAMDDA and Ordered FAM with n_{clust} = (Number of Classes) + 1.

Dataset	**FAM**					**Ordered FAM Gen.**	**Ordered FAMDDA Gen.**
	Worst Gen.	**Best Gen.**	**Avg. Gen.**	**std. dev.**	n_{clust}		
Breast	93.10	96.12	94.35	0.95	3	94.39	96.90
Glass	57.97	76.81	63.77	6.18	7	69.56	73.17

Table 3. Average network size for FAM and Ordered FAMDDA, and network size of Ordered FAM with n_{clust} = (Number of Classes) + 1.

Dataset	**FAM**		**Ordered**	**Ordered FAMDDA**	
	Avg. Net Size	n_{clus}	**Net Size**	**Avg. Net Size**	**Confidence Interval.**
Breast	8	3	9	7.3	[6.9, 7.7]
Glass	27	7	30	25.7	[23.4, 28.2]

5 The Circulating Water (CW) System

In this section, the application of Ordered FAMDDA to monitor the CW system in a power generation plant is described. As shown in Figure 2, the CW system includes all piping and equipment (such as condensers and drum strainer) between seawater intake and the outfall where water is returned to the sea. In the study we monitor the operating conditions of the condensers as the cooling water flowing through the system, with central attention focused on heat transfer problems, i.e., either good heat transfer or poor heat transfer, in the condensers.

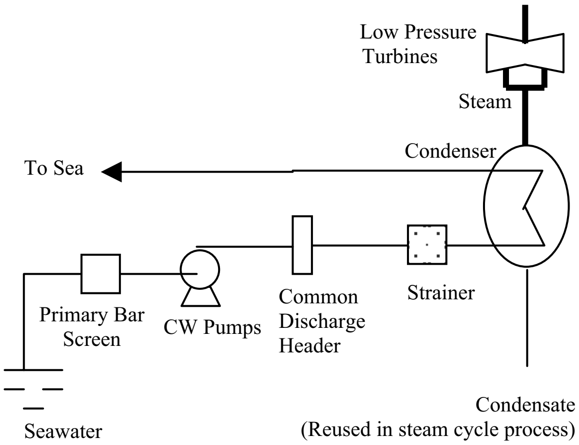


Figure 2. The Circulating Water system.

The data samples were collected on 80MW of the targeted power generation from several cycles of light up and shutdown periods of a power-generating unit. The database consisted of 2439 samples, with 1224 samples indicated poor heat transfer, and the remaining indicated good heat transfer in the condensers. Each sample data comprised temperature and pressure measurements at various inlet and outlet points of the condensers. Two-thirds of the data (i.e., 1626 samples) were selected for Ordered FAMDDA training and the remaining data (i.e. 813 samples) were used as unseen data in the test set. Note that the percentage of data from each class in the training set was in accordance with the percentage of each class in the entire dataset. The Ordered FAMDDA parameters used throughout the experiments were $n_{clust} = 3$, $\beta = 1$, $\varepsilon_{\min} = 0.1$, $\overline{\rho}_a = 0.0$. This was deemed as the "default" setting for Ordered FAMDDA which required no effort in fine-tuning the parameters. Ten independent experiments were conducted. The mean of accuracy rates and the network size in terms of number of nodes, and the associated 95% confidence intervals were computed using bootstrapping. The classification results of FAM and Ordered FAM were also obtained through similar experimental procedure and parameter settings aforementioned. All the results are summarised in Table 4. The generalisation performance of FAM, Ordered FAM. and Ordered FAMDDA ranged between 92.29–93.99%, 90.11–91.98%, and 93.51–94.93%, respectively, with the number of nodes ranged between 8.7–10.2, 5.6–7.2, 7.5–9.5, respectively. It is observed from Table 4 that the bootstrapped mean of Ordered FAMDDA (94.19%) is better than those of FAM (91.10%) and Ordered FAM's (93.09%). In addition, the generalisation performance of Ordered FAMDDA spans at a higher test accuracy interval than the FAM's and Ordered FAM's, respectively, with slight overlap and no overlap in the intervals. On the other hand, notice that the network size of Ordered FAMDDA spans in the interval between the intervals for FAM and Ordered FAM. In a nutshell, Ordered FAMDDA is a more efficient classifier than FAM and

Ordered FAM in this study at the expense of moderate network size that is between the network size of Ordered FAM and FAM.

Table 4. Comparisons the results of Ordered FAMDDA, Ordered FAM and FAM in monitoring the heat transfer condition of the CW system.

	Bootstrapped Accuracy (%)		**Bootstrapped #Nodes**	
	Mean	**Confidence Interval**	**Mean**	**Confidence Interval**
Ordered FAMDDA	94.19	[93.51, 94.93]	8.5	[7.5, 9.5]
Ordered FAM	91.10	[90.11, 91.98]	6.4	[5.6, 7.2]
FAM	93.09	[92.29, 93.99]	9.5	[8.7, 10.2]

6 Summary

In this paper, a novel adaptive conflict-resolving network, which integrates the ordering algorithm, FAM, and the DDA algorithm into a common framework, is described. The potential of the proposed Ordered FAMDDA network is demonstrated using two benchmark databases from the UCI machine learning repository and a real-world condition monitoring task in a power generation station. For the condition monitoring task, no hand-crafting of network parameters was performed. Indeed, it is our belief that a good system should require as few user-specified parameters as possible. The Ordered FAMDDA network has exhibited encouraging and promising performance in the two benchmark problems as well as the real-world condition monitoring task. Statistical analysis using bootstrapping has also been applied to ascertain the stability of the results. Further work will focus on a more comprehensive study on the effectiveness of Ordered FAMDDA in other classification tasks.

Acknowledgements

The authors are grateful to Prai Power Station, Tenaga Nasional Generation Sdn. Bhd, Penang, Malaysia, for the provision of the database used in this research work. The authors acknowledge the research grants provided by Ministry of Science, Technology, and Innovation Malaysia (No. 06-02-05-8002 & 04-02-05-0010) that have in part resulted in this article.

References

Berthold, M.R. and Diamond, J. (1995), "Boosting the performance of rbf networks with dynamic decay adjustment," in Tesauro, G., Touretzky, D.S., and Leen, T.K., editors, *Advances in Neural Information Processing Systems*, vol. 7, Cambridge, MA, MIT Press.

Blake, C. and Merz, C. (1998), UCI Repository of Machine Learning Databases, URL http://www.ics.uci.edu/~mlearn/MLRepository.html

Carpenter, G. and Grossberg, S. (1987), "A massively parallel architecture for a self-organizing neural pattern recognition machine," *Computer Vision, Graphics and Image Processing*, vol. 37, pp. 54-115.

Carpenter, G., Grossberg, S., and Rosen, D. (1991), "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural Networks*, vol. 4, pp. 759-771.

Carpenter, G., Grossberg, S., Markuzon, N., Reynolds, J., and Rosen, D., (1992), "Fuzzy ARTMAP: A neural network architecture for incremental learning of analog multidimensional maps," *IEEE Trans. Neural Networks*, vol. 3, pp. 698-713.

Dagher, I., Georgiopoulos, M., Heileman, G.L., and Bebis, G. (1999), "An ordering algorithm for pattern presentation in fuzzy ARTMAP that tends to improve generalization performance," *IEEE Trans. Neural Networks*, vol. 10, pp. 768-778.

Efron, B. (1979), "Bootstrap methods: another look at the Jackknife," *The Annals of Statistics*, vol. 7, pp. 1-26.

Grossberg, S. (1976), "Adaptive pattern recognition and universal recoding ii: feedback, expectation, olfaction, and illusions," *Biological Cybernectics*, vol. 23, pp.187-202.

Huber, K.-P. and Berthold, M.R. (1995), "Building precise classifiers with automatic rule extraction**,"** Proceedings of the IEEE Int. Conf. Neural Networks, vol. 3, pp. 1263-1268.

Lim, C.P. and Harrison, R.F. (1997), "An incremental adaptive network for on-line supervised learning and probability estimation," *Neural Networks*, vol. 10, pp. 925-939.

Mangasarian, O.L. and Wolberg, W.H. (1990), "Cancer diagnosis via linear programming," *SIAM News*, vol. 23, pp. 1-18.

Moody, M.J. and Darken, C.J. (1989), "Fast learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, pp. 281-294.

Murphy, P. and Ana, D. (1994), UCI Repository of Machine Learning Databases, Dept. Comput. Sci. Univ. California, Irvine, CA, Tech. Rep.. Available http://www.ics.uci.edu/~mlearn/MLRepository.html

Riedmiller, M. and Braun, H. (1993), "A direct adaptive method for faster backprogation learning: the RPROP algorithm," Proceedings of the IEEE Int. Conf. Neural Networks, vol. 1, pp. 586-591.

Simpson, P.K. (1992), "Fuzzy min-max neural networks – Part 1: Classification," *IEEE Trans. Neural Networks*, vol. 3, pp. 776-786.

System Description and Operating Procedures (1999), Prai Power Station Stage 3, vol. 14.

Tou, J.T. and Gonzalez, R.C. (1976), *Pattern Recognition Principles*, Reading, MA: Addison-Wesley.

A Method to Enhance the ‘Possibilistic C-Means with Repulsion’ Algorithm based on Cluster Validity Index

Juan Wachs[1], Oren Shapira[1] and Helman Stern [1]

{juan|orensha|helman}@bgu.ac.il,
[1]Department of Industrial Engineering and Management, Intelligent Systems Division, Ben-Gurion University of the Negev, Be’er-Sheeva , Israel 84105

Abstract: In this paper, we examine the performance of fuzzy clustering algorithms as the major technique in pattern recognition. Both possibilistic and probabilistic approaches are explored. While the Possibilistic C-Means (PCM) has been shown to be advantageous over Fuzzy C-Means (FCM) in noisy environments, it has been reported that the PCM has an undesirable tendency to produce coincident clusters. Recently, an extension of the PCM has been presented by Timm et al., by introducing a repulsion term. This approach combines the partitioning property of the FCM with the robust noise insensibility of the PCM. We illustrate the advantages of both the possibilistic and probabilistic families of algorithms with several examples and discuss the PCM with cluster repulsion. We provide a cluster valid-ity function evaluation algorithm to solve the problem of parameter optimization. The algorithm is especially useful for the unsupervised case, when labeled data is unavailable.

Keywords: Possibilistic and probabilistic fuzzy clustering; Fuzzy C-Means; Cluster validity index, Robust methods.

1 Introduction

Cluster analysis is the process of classifying objects into subsets that have meaning in the context of a particular problem. The objects are thereby organized into an efficient representation that characterizes the population being sampled [1]. Hard clustering methods assume that each observation belongs to one class, however in practice clusters may overlap, and data vectors belong partially to several clusters. This scenario can be modeled properly using the fuzzy set theory [2], in which the membership degree of a vector x_k to the i-th cluster (u_{ik}) is a value from [0,1] interval. Bezdek [3] explicitly formulated this approach oriented to clustering by introducing the Fuzzy c-mean clustering algorithm. Unfortunately, this method showed the difficulty of high sensibility to noises and outliers in the data. To reduce this undesirable effect, a number of approaches have been proposed, but the most remarkable has been the possibilistic, introduced by [4], with their possibilistic c-means algorithm. In this algorithm the membership is interpreted as the compatibilities of the datum to the class prototypes (typicalities) which correspond to the intuitive concept of degree of belonging or compatibility. In the case of poor initializations, it is possible that the PCM will converge to a "worthless" partition where part or all the clusters are identical (coincident) while other clusters go undetected. Recently, a new scheme has been proposed, in order to overcome the problem of cluster mutual attraction forces, by introducing a supplementary term for cluster repulsion [5]. By use of cluster repulsion, as good separation between clusters is obtained, as with the FCM, while keeping the intuitive concept and the noise insensibility introduced by the PCM. The goal of this paper is to establish a connection between the possibilistic approach and cluster validation indices, such that the quantitative superiority of the 'PCM with repulsion' over other methods is tangible.

The organization of this paper is as follows. In Section 2, we analyze the possibilistic approach by [4] developed to cope with the problem of noise and concept of compatibility, but lacks of clusters discrimination. In Section 3, we review the recently proposed method by [5] based on repulsion between clusters, and reports the difficulty of choosing the proper value of the weighting factor γ. In Section 4, we compare four clustering techniques using several datasets and suggest a graphical method to obtain the optimal weighting factor γ; Finally, Section 5 presents our summary and conclusions.

2. Possibilistic Fuzzy Clustering

The most widely used prototype-based clustering method for data partition is probably the ISODATA or Fuzzy C-Means (FCM) algorithm [3]. Given a set of n data patterns, $X = x_1 ,...., x_k ,..., x_n.$, the algorithm minimizes a weighted within group sum of squared error objective function. A constraint assures relative numbers for the membership, and therefore is not suitable for applications where memberships are supposed to represent typicalities or compatibilities. Thus, in the FCM the memberships in a given cluster of two points that are equidistant from the prototype of the cluster can be significantly different and memberships of two points in a given cluster can be equal even thought the two points are arbitrarily far away from each other [6]. This situation is illustrated in Figure 1.

In this example, there are two clusters and a pair of points x,y; which represents an outlier and a noise point respectively.

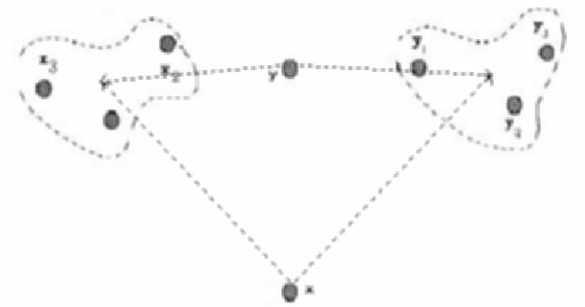


Figure 1. Example of dataset with a noise and outlier points

Both points x and y will be assigned a membership value of 0.5 to both clusters by the FCM. One concludes that this membership values are unrepresentative of the degree of 'belonging', but also they cannot discriminate between an outlier datum and noise.

The PCM formulation relaxes the objective function of the FCM by dropping the sum to 1 constraint. To avoid a trivial solution of $u_{ik} = 0$ for all i, a penalty term is added which forces u_{ik} to be as large as possible, by modifying its objective function as follows:

$$J(U,V) = \sum\nolimits_{k=1}^{n} \sum\nolimits_{i=1}^{c} u_{ik}^{m} d^2(x_k, v_i) + \sum\nolimits_{i=1}^{c} \eta_i \sum\nolimits_{k=1}^{n} (1 - u_{ik})^m \qquad (1)$$

Where η_i is a positive number, and $u_{ik} \in [0,1]$. The new cluster centers v_i and the membership update equations are:

$$u_{ik} = \frac{1}{1 + \left[\dfrac{d^2(x_k, v_i)}{\eta_i} \right]^{1/(m-1)}} \quad , v_i = \frac{\sum\nolimits_{k=1}^{n} u_{ik}^{m} X_k}{\sum\nolimits_{k=1}^{n} u_{ik}^{m}} \qquad (2)$$

The parameter η_i is evaluated for each cluster separately; it determines the distance at which the membership degree equals 0.5.

$$\eta_i = K \sum\nolimits_{k=1}^{n} u_{ik}^{m} d^2(x_k, v_i) \Big/ \sum\nolimits_{k=1}^{n} u_{ik}^{m} \qquad (3)$$

Using (3) η_i is proportional to the average fuzzy intracluster distance of cluster v_i. Usually K is chosen to be 1.

The main drawback of this promising approach appears when the objective function in (1) is truly minimized, and this occurs when all cluster centers are identical (coincident centroids). This failure is due to the reason that the membership degrees (2) depend only on the distance between the point to the cluster, and not on its relative distance to other clusters. Usually only part of the centroids are coincident, since the algorithm converge in a local minimum of the objective function (1), still this is an undesirable behavior for a clustering algorithm [7].

3. Possibilistic Fuzzy Clustering with Repulsion

Recently, the Possibilistic Fuzzy Clustering with Repulsion was proposed to address the drawbacks associated with the FCM and the PCM. This method aims to minimize the intracluster distances [8], while maximizing the intercluster distances, without using implicitly the 'sum 1 restriction', but by adding a cluster repulsion term to the objective function (1).

$$J(U,V)=\sum_{k=1}^{n}\sum_{i=1}^{c}u_{ik}^{m}d^{2}(x_k,v_i)+\sum_{i=1}^{c}\eta_i\sum_{k=1}^{n}(1-u_{ik})^{m}+\gamma\sum_{i=1}^{c}\sum_{k=1,k\neq i}^{c}\frac{1}{d^{2}(v_i,v_k)} \quad (4)$$

Where γ is a weighting factor, and u_{ik} satisfies:

$$u_{ik}\in[0,1] \qquad ,\forall i \quad (5)$$

The repulsion term is relevant if the clusters are close enough. With growing distance it becomes smaller until it is compensated by the attraction of the clusters. On the other hand, if the clusters are sufficiently spread out, and the intercluster distance decreases (due to the first two terms), the attraction of the cluster can be compensated only by the repulsion term.

Minimization of (4) w.r.t. to cluster prototypes leads to:

$$v_i=\frac{\sum_{j=1}^{n}u_{ij}x_j-\gamma\sum_{k=1,k\neq i}v_k\frac{1}{d^{2}(v_k,v_i)}}{\sum_{j=1}^{n}u_{ij}-\gamma\sum_{k=1,k\neq i}\frac{1}{d^{2}(v_k,v_i)}} \quad (6)$$

Singularity occurs when one or more of the distances $d_2(v_k, v_i) = 0$ at any iterate. In such a case, (6) cannot be calculated. When this happens, assign zeros to each nonsingular class (all the classes except i) and assign 1 to class i, in the membership matrix U. Similar as for the PCM algorithm, the formula for updating the membership degrees u_{ik} is obtained using (2). The weighting factor γ is used to balance the attraction and repulsion forces, i.e., minimizing the intradistances inside clusters and maximizing the interdistances between clusters. The central problem of this algorithm is that it requires a resolution parameter γ, and no clue is given about the correct range of this parameter [9].

Cluster validity studies the "goodness" of a partition generated by a clustering algorithm. The sum of intracluster distances, over the minimum of the intercluster distances is one of the most commonly used validity measures because of its analytical simplicity. This formulation is known as the Xie-Beni index v_{XB}, and is defined as:

$$v_{XB}(U,V;X)=\frac{\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}^{2}\,\|x_k-v_i\|^{2}}{n(\min_{i\neq j}\{\|v_i-v_j\|^{2}\})} \quad (7)$$

A good (U,V) pair should produce a small value of (7) because u_{ik} is expected to be high when $\|x_k-v_i\|$ and well-separated v_i's will produce a high value in the denominator of (7). Consequently, the minimum of v_{XB} , is the most desirable partition.

4. Tests Examples

The first example illustrates two well-separated noise-free clusters of 30 points, drawn from two components of a normal distribution, with μ_{1x}=2, μ_{1y}=3, μ_{2x}=5, μ_{2y}=3 and σ=0.5, see Figure 2.a. In Figure 2.b, the crisp partition for the four algorithms are similar, the centroids are almost the same. Each point is assigned to the cluster which it has the highest membership for the FCM, and for the possibilistic algorithms, the highest typicality was used. Ties are broken arbitrary. After adding 28 points of random noise to the cluster on the right of Figure 1.a with μ_{2x}=6.5, μ_{2y}=4.5 and σ=1, the crisp partition presents significant differences; the FCM and FPCM performs the worst, while the other two present similar partitions, see Figures 2.c - 2.f.

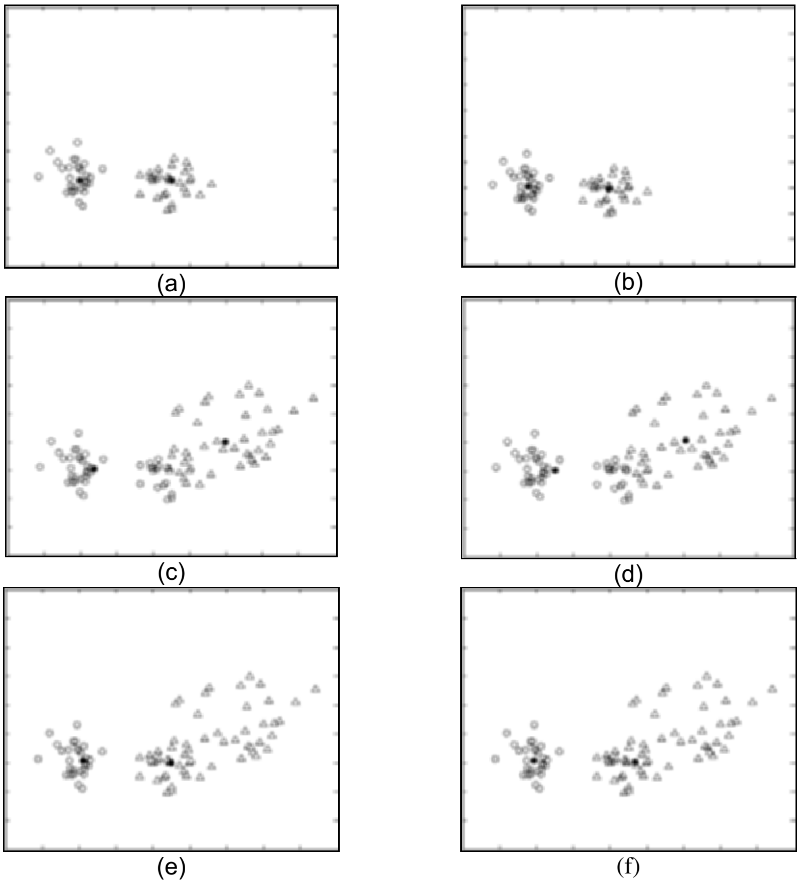


Figure 2. Partition of the synth. data set: (a) original data; (b) FCM, FPCM, PCM and ‘PCM with rep.’ partition; (c) FCM partition with noise; (d) FPCM partition with noise; (e) PCM partition with noise; (f) ‘PCM with rep.’

The FCM algorithm was used to obtain a good initialization for the PCM. The noise addition to the original data affected significantly the cluster centers in the probabilistic based algorithms, while in the possibilistic based algorithms, the cluster centers are virtually unchanged. The parameters used were: m=2, c=2, ε=0.0001, η=2, γ=10.

Table 1, shows the centroids of the clusters, as seen by the different four algorithms. The last column presents the deviation of the centroids from their original location, after adding the noisy data. The performance of the PCM and 'PCM with repulsion' are acceptable. The FCM algorithm gave the poorest estimates for the centroids, since its deviation is the highest.

The second example shows a more realistic example with the well-known IRIS data set [10]. IRIS consists of 150 points in four dimensions that represent three physical classes each with 50 points.

The numerical representation of two classes has substantial overlap, while the third is well separated of the other two; therefore a three clusters classification is recommended. The algorithms discussed above have been tested on the IRIS data set, only both attributes, petal length and petal width have been used, since they carry the most information about the distribution of the iris flowers. Several runs of the four algorithms on IRIS data set were made, with different initializations and different (m, η, γ) tuples. Here we report results only for the initialization in Table 2. The performance of each algorithm (classification accuracy) is measured as the ratio between numbers of correct classification, and the total sample set. The number of mistakes is based on comparing the hardened version of the membership and typicalities matrices, to the physically correct crisp 3-partition of IRIS.

Table 1. Centroids estimation using the FCM, PCM, FPCM, and 'PCM with Repulsion' for Figure 2

	Fuzzy c-Means	Possibilistic	Mixed c-Means	Possiblistic with Repulsion
Original Data Set	(1.89 3.08) (4.47 2.91)	(1.96 3.04) (4.41 2.93)	(1.89 3.08) (4.47 2.91)	(2.01 3.06) (4.37 2.93)
With Noise	(2.49 3.02) (6.08 4.06)	(2.10 3.05) (4.48 2.98)	(2.29 3.04) (5.90 3.94)	(1.92 3.06) (4.69 3.01)
Deviation	2.06	0.16	1.81	0.34

Table 2. Classification accuracy on the IRIS data using the FCM, PCM, FPCM, and 'PCM with Repulsion'

Parameters			Accuracy				Iterations			
m	η	γ	FCM	FPCM	PCM	rep.PCM	FCM	FPCM	PCM	rep. PCM
2	3	0.1	82%	82%	64%	53%	26	12	26	13
		1				78%				
		15				97%				
		50				89%				
		100				76%				
		200				62%				

From Table 2, we find that the FPCM and 'PCM with repulsion' give higher or same accuracies than FCM for best γ cases, which indicate that typicality based *classification represents better the physical data-partition, than membership values. Note that typicality-based classification accuracy failed for the PCM case, where two cluster coincidences affected the performance of the algorithm. The 'PCM with repulsion', using a good selection of parameter γ, shows significant superiority over the other algorithms studied here. Fortunately, Table 2, also shows that the number of iterations required for the PCM 'with repulsion' is similar to the FPCM, and is about half of that of the PCM and FCM.

Clusters detected by the PCM as a function of gamma, using m=2, are depicted in Figure 3. For γ=0.1 only two clusters are detected because the possibilistic algorithm is not forced to divide the data, and both clusters are coincident (Figure 3.a). By incrementing γ the attraction between clusters, decreases and the centroids of the coincident clusters are driven apart. For γ=1, the distance between samples assigned to clusters and their respective centroid is minimized, and that the distance between clusters is maximized, (Figure 3.b, Figure 3.c). For further values of γ, the repulsion increases with the distance of the clusters, driving them ever farther apart, hence harming the classification accuracy (Figure 3.d).

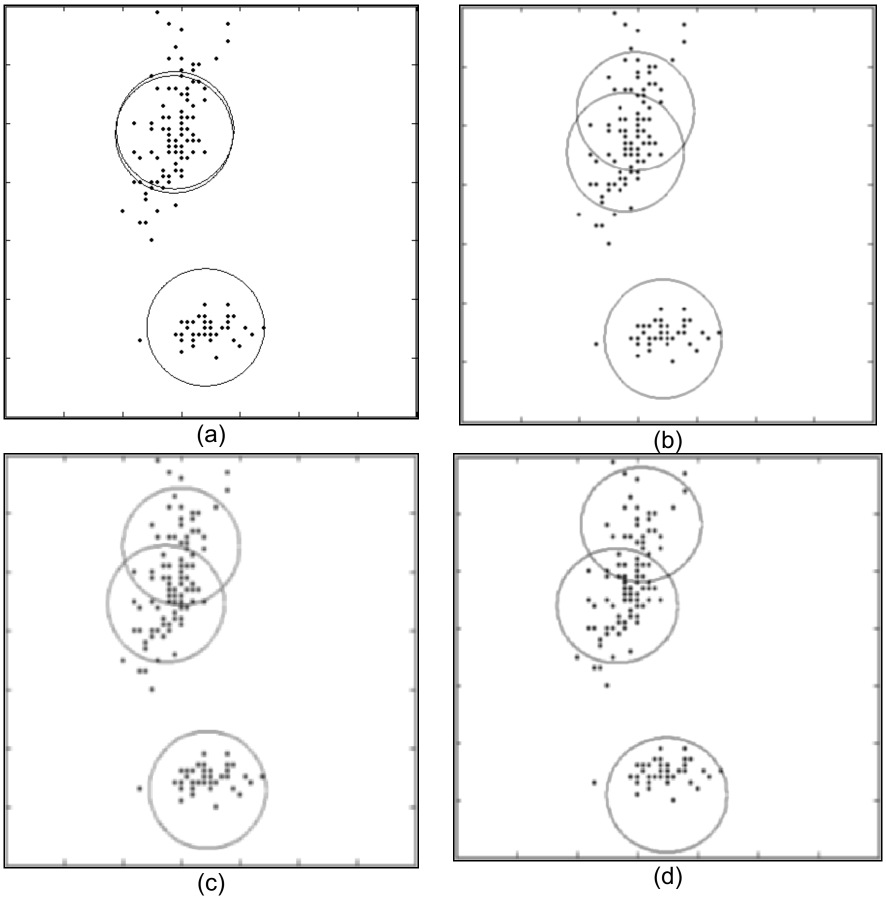
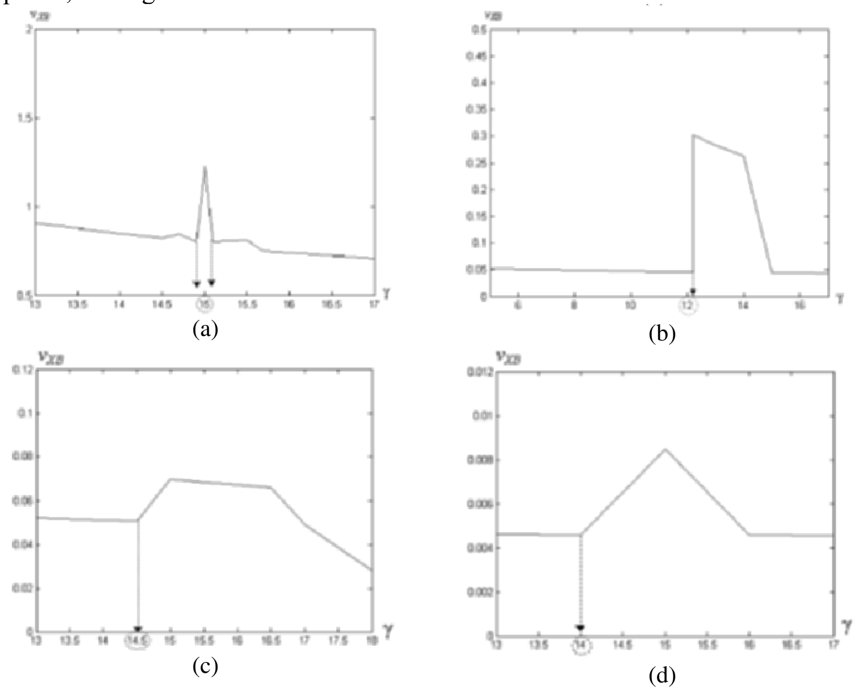


Figure 3. Iris dataset classified using 'PCM with repulsion': (a) γ=0.1, (b) γ=1, (c) γ=20, (d) γ=40.

As shown in Table 2, in the ‘rep. PCM’ column, the parameter γ affect the performance of the data partition, and therefore, cluster validity measures should give some clue about the optimal value of γ.

For an unsupervised data set it is not possible to use the classification accuracy measure, and therefore an alternative method must be employed to find the best value of the weighting factor γ. We note that for the optimal value of γ, all the clusters are overall compact, and separate to each other. A local minima in the compactness and separation validity function (7) indicates a better partition of the data set. For small values of γ, coincident centroids appear, then the denominator of (7) is negligible, and as a result of this, the v_{XB} is very high. However it, is noted that v_{XB} as a function of γ, is monotonically decreasing when γ gets very large (the denominator of (7) grows to infinity). Nevertheless, a close-up of the v_{XB} as a function shows that a local minimum is reached when the centroid is placed at its optimal value, see Figure 4.a. Still, if the centroid is misplaced far enough from all the others centroids, the points that should belong to clusters will be assigned to the closer centroid, and hence v_{XB} will drop abruptly. We conclude that the use of the Xie-Beni validation index is valid only around the optimal values for the centroids, where the validation function behaves convex. Through plotting the function v_{XB} against γ, the optimum of gamma occurs at the side of the “peak”, see Figure 4.



(a)

(b)

(c)

(d)

Figure 4. Plots of v_{XB} vs. : The monotonic decreasing function, a close-up peak for: (a) Iris data, (b) synthetic data without noise (c) synthetic data with noise and (d) Wisconsin cancer breast data.

All the γ values immediately before and after the peak are candidates to be optimal. The γ^* (optimal weighting factor) is the one which yields the lower v_{XB} between all the candidates. Figure 4.b, shows the plots for the hard partition of IRIS obtained using the 'PCM with repulsion', with m=2. The figure illustrates two valleys around the peak in γ=15, where the optimal between them is γ^*=15.1, since there the Xie-Beni index is the lowest, v_{XB}^* =0.8. Observing Table 2, we see that optimal classification accuracy (97%) was obtained for γ=15, therefore the result obtained graphically is satisfactory. This approach has been tested on the synthetic data set with and without noise, and we report results of 100% of classification accuracy using γ^*=12 and γ^*=15 respectively, see Figure 4.b-c. The last test has been conducted on the Wisconsin Breast Cancer Data [11] and a 95% of classification accuracy was obtained using γ^*=14.9, see Figure 4.c. Once we have defined a method based on the validity function v_{XB}, our implementing strategy can be summarized into the following pseudo algorithm:

Algorithm 1 –Find Optimal Gamma

```
1)  **Initialize** $\gamma \leftarrow 0.2$, $\Delta \leftarrow 0.1$, $m \leftarrow 2$, $K \leftarrow 1$, $v_{XB}(U,V;X;\gamma-\Delta) \leftarrow 0$;
2)  **Initialize** centroids $v_i$ randomly;
3)  Use (2) to **update** typicalities $u_{ik}$ and to **update** centroids;
4)  **Calculate** $\eta_i$ using (3);
5)  **Do** converge test; if negative goto 3;
6)  **Compute** function $v_{XB}(U,V;X;\gamma)$;
7)  **Repeat** steps 2,3,4,5 for $v_{XB}(U,V;X;\gamma+\Delta)$;
8)  **If** $v_{XB}(U,V;X;\gamma-\Delta) > v_{XB}(U,V;X;\gamma) < v_{XB}(U,V;X;\gamma+\Delta)$ **then** $\gamma_k=\gamma$;
9)  **If** $\gamma$=stop_value **then stop**; **else** $\gamma = \gamma+2\Delta$;
10) **Goto** 2;
11) **Find** $\gamma^* = min\ v_{XB}(U,V;X;\gamma_k)$ over all $k$;
```

Steps 2,3,4 and 5 are the 'PCM with repulsion' algorithm. The convergence test is $||J_{i+1} - J_i|| < \varepsilon$ where i is the number of iteration, J is the cost function obtained from (4) and ε is the accepted error margin. The fuzziness parameter m usually is set to 2, and the number of clusters c is fixed a priori. Initial values of v_i are datum selected randomly from the sample set. This method is computationally expensive in high dimensions however its performance can be improved by considering the shapes of clusters and a better validity index.

5. Conclusions

We have reviewed four algorithms for possibilistic and probabilistic clustering: FCM, PCM, FPCM, and the 'PCM with repulsion'. We then proposed a functional evaluation method and a strategy to obtain a good range for the weighting factor γ, based on the Xie and Beni index validation measure. Computational examples on three data sets were used to compare the four algorithms described in this paper, and to support the assertion that the weighting factor obtained by our function evaluation algorithm is useful. In summary, the PCM in its original form is not very suitable for clustering due to the undesirable coincident centroids, but it provides robustness to noise and an intuitive interpretation of the membership values. The 'PCM with repulsion' overcomes this problem, and showed a satisfactory performance when a proper value for the weighting factor γ was used. Our study has shown a correspondence between the Xie-Beni validity function and the range of the weighting factor of γ. The points obtained using the validity index for discrete γ is a quantization of all the points on the curve of the validity function, and therefore the "peaks" might be missed if to coarse a quantization is used. Although the method proposed is computationally intensive, it is still better than supervising the clustering algorithm, especially when labeling knowledge is unavailable.

Further investigation and more numerical tests are required before much can be asserted about an optimal range for the weighting factor γ, in the presence of noisy environments, which may cause false "peaks" in the validity function curve. The Xie-Beni index only measures compact and separate clusters, therefore the reliability of the approach may be improved by considering better validity measures.

References

[1] Jain, A. K. and Dubes, R. C. (1988), Algorithms for clustering data, *Prentice Hall*, New Jersey.

[2] Zadeh, L. A. (1965), "Fuzzy Sets," Information and Control, vol. 8, pp. 338-353.

[3] Bezdek, J. C., (1982), Pattern recognition with fuzzy objective function algorithms, *Plenum Press*, New York.

[4] Krishnapuram, R. and Keller, J. (1993), "A possibilistic approach to clustering," *IEEE Trans. Fuzzy Systems*, vol. 1, pp. 98-110.

[5] Timm, H., Borgelt, C. and Kruse, R. (2001), "Fuzzy cluster analysis with cluster repulsion," *In Proc. of the European Symp. on Intelligent Tech., Hybrid Syst. and their implementation on Smart Adapt. Syst.*, Tenerife, Spain.

[6] Krishnapuram, R. and Keller, J. (1996), "The Possibilistic C-Means algorithm: Insights and recommendations*", IEEE Trans. Fuzzy Systems*, vol. 4, pp. 385-393.

[7] Barni, M., Cappellini V., and Mecocci, A. (1996), "Comments on 'A Possibilistic Approach to Clustering'," *IEEE Trans. Fuzzy Systems*, vol. 4, pp. 393-396.

[8] Bezdek, J. C. and Pal, N. R. (1998), "Some New Indexes of Cluster Validity," *IEEE Trans. on SMC, Part B*, vol. 28, no.3, pp. 301-315.

[9] Dave, R. N. and Krishnapuram, R. (1997), "Robust clustering methods: a unified view," *IEEE Trans. Fuzzy Systems*, vol. 5 no. 2, pp.270-293.

[10] Fisher, R. (1936), “The use of multiple measurements in taxonomic problems,” Annals of Eugenics, vol.7, no. 2, 179-188.

[11] Merz, C. J. and Murphy, P. M. (1996), UCI Repository of Machine Learning Databases, http://www.ics.uci.edu/~mlearn/MLRepository.html, University of California, Department of Information and computer Science.

Part III

Optimization

Design Centering and Tolerancing with Utilization of Evolutionary Techniques

Lukasz Zielinski, Jerzy Rutkowski

Silesian University of Technology, Institute of Electronics
ul. Akademicka 16, 44-100 Gliwice, Poland
lukasz.zielinski@comverse.com, jr@boss.iele.polsl.gliwice.pl

Summary. This paper undertakes the problem of Design Centering and Tolerancing (DCT). This subject plays important role in design of analog circuits. Practically it is impossible to obtain the optimum solution by mathematically described computational methods, and the problem is numbered to NP-difficult category. Optimum nominal values and tolerances of analog circuit parameters such as resistance, capacitance and inductance, are determined based on evolutionary strategy. The approach is verified on benchmark example and significant improvement, when compared with other design methods, can be observed.

Key words: analog circuit, design centering, design tolerancing, evolutionary techniques, optimization

1 Introduction

Design Centering and Tolerancing (DCT) is one of the most important problem of computer-aided circuit design. Due to fabrication deviations of circuit parameters from nominal values, it is not possible to achieve a situation in which all fabricated circuits meet design specifications. It results in a production yield [4] being always less than 100%. There are several methods to improve this output and DCT is perceived as one of the most effective and comprehensive. It is assumed that for design specifications both initial nominal point coordinates and design tolerances are known. Then, selection of a circuit new nominal point and corresponding tolerances, such that the maximum possible yield is obtained, is the DCT aim [6]. So far, researchers have presented different approaches to DCT optimization. They will be discussed briefly in the next section. All the methods are based on classical optimization techniques. Evolutionary techniques [12] introduced in the 70's, originated from Darwin's evolution theory of natural choice and selection, have appeared to be a very effective tool in various optimization problems. Therefore, their use to DCT optimization seems reasonable. A new approach based on evolutionary strategy is proposed. After the method description, benchmark computational examples will be presented and the obtained results compared with those obtained by means of other design methods. This comparison shows significant increase of the yield when evolutionary optimization is applied. Results are also verified against product cost model, introduced in [1].

2 Design Centering and Tolerancing Methods

Two different approaches to DCT can be distinguished: Statistical approach and Geometrical approach. The combined approach has been proposed recently.

A. Statistical approach

The DCT is based on Monte Carlo analysis of the examined circuit. The analysis results are then investigated and optimum solution (nominal point and corresponding tolerances) is selected. Monte Carlo based methods are considered to be computationally expensive but on the other hand, the obtained solution is very accurate. The detailed description of statistical methods can be found in [6, 10]. In [3], statistical design with parametric sampling concept is presented, where an attempt to reduce the MC analysis effort is proposed.

B. Geometrical approach

This approach is based on mapping of feasible region i.e. region designated by design specifications, from specification space to parameter space. The obtained approximation in parameter space is called acceptability region and it is usually assumed that this region is convex [11, 13]. There are different methods [2, 5, 7] of such approximation and simplical approximation [8] is the most popular one. Then, centre of acceptability region is searched for and accepted as the DCT problem solution. Geometrical approach has some drawbacks. Selection of geometrical body that accurately approximates the region is one of them. Furthermore, calculations in parameter space of large dimension become enormously complex at both stages: mapping stage and centering stage.

3 New Method Description

It is assumed that an examined circuit is characterized by S design specifications and P design parameters, nominal values and tolerances $X = [x_1, tol_1, \ldots, x_P, tol_P]$. These initial nominal values of parameters as well as their tolerances are given by the design engineer. Normal distribution (1) within tolerance region is assumed.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot exp^{-\frac{(x-\delta)^2}{2\sigma^2}} \tag{1}$$

where, σ^2 is variance and δ is the expected value (nominal point). The production yield can be then expressed by (2).

$$Y(X) = \int_R f(X)dX \tag{2}$$

where, R is an acceptability region defined as a region in parameter space where all points satisfy S design constraints [6]. For the given nominal point and tolerances, the yield can be obtained by means of Monte Carlo analysis [10]. The aim of DCT is to optimally select new nominal values and corresponding tolerances, to enhance production yield $Y(X)$. This optimization is performed by means of evolutionary strategy. This evolutionary technique seems to be the most suitable for real number vector optimization. In all computations, self-adaptation of mutation step size and $(\mu + \lambda)$ survivor selection type have been assumed, where μ denotes size of parent population, λ denotes size of offspring population and a new population is created from both populations [14]. Chromosome consists of $2 \cdot P$ genes:

$$Chrom = [x_1, tol_1, \ldots, x_P, tol_P] \tag{3}$$

In each evolutionary cycle (iteration), from the initial parent population, through genetic operations, such as mutation and recombination, offspring population is created. Values of genes related to tolerances, can be limited by upper b_u and lower boundary b_l, suitably to design requirements. For each chromosome of both populations its fitness is calculated as the corresponding production yield (4).

$$Fit_i = a \cdot Y_i(X) + b \cdot \frac{\sum\limits_{p=1}^{P} tol_{xp}}{P \cdot b_u}; i = 1, \ldots, N = \mu + \lambda \tag{4}$$

Finally, best μ chromosomes create the new initial (parent) population and evolutionary cycle is repeated until the assumed stop condition is reached. This condition can be designated by the maximum iteration number $j = It_{max}$ or distance between maximum fitness of the current j^{th} iteration and the previous $(j-1)^{th}$ iteration:

$$|maxFit(j) - maxFit(j-1)| < \varepsilon \tag{5}$$

As can be seen, fitness function (4) consists of two factors: yield factor represented by weigh a and tolerance factor represented by weigh b, $a + b = 1$. By selecting the appropriate weighs it can be decided which part is prevailing in the function. Fitness of the best chromosome should be very close 1. A competitive product must address factors such as cost, performance or time-to-market, and quality. The importance of these factors will vary from product to product and market to market. Numerous product cost models are possible [15]. To additionally compare the obtained results, throw-away model (6) has been chosen. Its objective is to minimize the cost C_s of units satisfying all requirements.

$$C_s = \frac{C_u}{Y} \tag{6}$$

where, C_u is per unit manufacturing cost (component costs and fixed costs e.g. printed circuit board cost, test labor cost) and Y is the yield. It is clear, that circuits of nominal component values and their tolerances representing minimum on the cost versus yield are preferred.

4 Computational Examples

The method will be illustrated by two practical examples, originally considered in [1, 9]. Evolutionary strategy parameters for all examples are: $\mu = 20$, $\lambda = 80$, $It_{max} = 100$, number of Monte Carlo simulations per single chromosome (individual) $MC = 100$. These parameters selection has been based on engineer's experience and is outside the scope of this paper. Additionally, tolerance boundaries: $b_l = 1$, $b_u = 20$ in the first example and $b_l = 1$, $b_u = 10$ in the second have been assumed.

A. Example 1

The practical example, high-pass filter originally discussed in [9], is reproduced in Figure 1. Design specifications are presented in Table 1, where $\omega_0 = 990$Hz and relative attenuation in [dB] is described by (7).

$$\bar{I}_{loss}(j\omega) = 20 \log |\frac{V_1}{V_2}(j\omega)| - 20 \log |\frac{V_1}{V_2}(j\omega_0)| \tag{7}$$

Design circuit parameters are considered to be: L1, L2, C3, C4, C5, C6, C7 with initial nominal values [3.85H 3.18H 11.80nF 8.73nF 10.45nF 39.04nF 90.13nF].

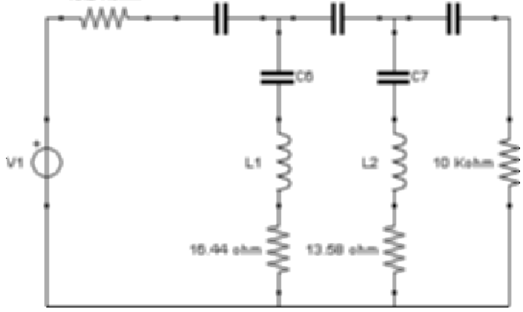


Fig. 1. High-pass filter example

The cost function for this example is taken as (8) and is exactly the same as chosen in [1]. It allows to obtain better and more exact comparison of the methods.

$$C_S = \frac{C_A + 2\sum_{i=1}^{2} \frac{1}{t_i} + \sum_{i=3}^{7} \frac{1}{t_i}}{Y} \tag{8}$$

$C_A = 250$ represents the fixed part of C_u as defined in (6). The nominal values corresponds to the mean values tolerances t_i express correlation between mean (nominal) values δ_i and standard deviations σ_i , as described in (9) [1].

$$t_i = \frac{3\sigma_i}{\delta_i} \tag{9}$$

The yield before DCT optimization has been computed and for the assumed *tol* = 15% (same for all parameters) $Y(X) = 50.66\%$. The strategy has resulted in 7 equivalently good solutions with 98.66% yield, where $a = 0.5$, $b = 0.5$. It is worth mentioning that these solutions have been already found in the 23^{th} iteration. Next, prevalence of the yield factor has been assumed: $a = 0.95$, $b = 0.05$. The yield has increased to 100% and also several results have been obtained. Only minor increase of the cost has been observed. Table 2 presents only one the best solution for each case.

Table 1. Specifications for high-pass filter

Frequency (Hz)	$\bar{I}_{loss}$(dB)
170	> 45
350	> 49
440	> 42
630	$-0.05 < \bar{I}_{loss} < 4.0$
650, 720,740,760, 940, 1040,1800	$-0.05 < \bar{I}_{loss} < 1.75$

B. Example 2

Figure 2 shows low-pass 11^{th} order Chebychev filter, originally considered in [1]. In Table 3 filter design specification is presented, where $\omega_0 = 0$Hz and relative attenuation in [dB] is computed from (6).

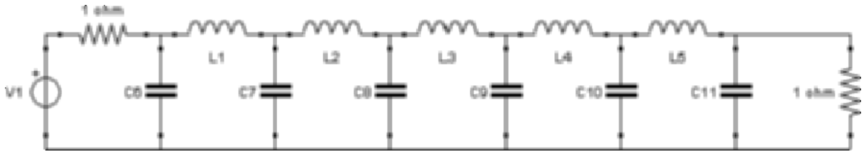


Fig. 2. Low-pass filter example

Table 2. Results of example 1

Component	Evolutionary Strategy $a = 0.5$ $b = 0.5$		Evolutionary Strategy $a = 0.95$ $b = 0.05$		Parametric sampling with 300 point data base		Statistical design (worst-case centered design)		Kjellstrom's design with 2000 analysis	
	Δ[%]	*tol*[%]	Δ[%]	*tol*[%]	Δ[%]	*tol*[%]	Δ[%]	*tol*[%]	Δ[%]	*tol*[%]
L1	-15.02	27.15	-18.52	3.02	1.36	8.87	5.38	4.48	-5.04	9.3
L2	6.41	8.52	2.26	4.4	-2.93	11.85	-16.82	4.32	-8.34	9.66
C3	19.54	6.19	16.51	16.08	1.11	9.33	4.83	5.23	5.48	11.04
C4	3.96	12.66	7.26	7.51	1.43	8.00	19.7	3.96	11.16	9.57
C5	24.49	19.06	28.62	14.17	6.52	9.39	39.43	4.25	24.46	9.63
C6	14.1	9.95	26.86	1.92	-4.2	9.27	-13.09	4.05	2.4	11.25
C7	-16.23	6.34	-19.89	17.72	5.04	14.00	7.77	5.24	-2.96	20.31
Fitness value	**0.5**		**0.99**		**-**		**-**		**-**	
Yield Y	**98.66**		**100**		**97.33**		**100**		**96.33**	
Cost C_s	**3.81**		**3.89**		**3.5**		**4.53**		**3.49**	

There are eleven design parameters taken into account: L1, L2, L3, L4, L5, C6, C7, C8, C9, C10, C11. Their nominal values are [0.2251H 0.2494H 0.2523H 0.2494H 0.2251H 0.2149F 0.3636F 0.3761F 0.3761F 0.3636F 0.2149F]. For these parameters *tol* = 10%, the yield has been calculated, by means of Monte Carlo analysis. The cost function of the following form is proposed:

$$C_s = \frac{\sum\limits_{i=1}^{11} \frac{1}{t_i}}{Y} \qquad (10)$$

where, t_i expresses correlation between standard deviation and mean value, calculated from (9). Table 4 summarizes all optimal nominal values and tolerances obtained by the ES, for $a = 0.5$, $b = 0.5$ and $a = 0.9$, $b = 0.1$. For each case several equivalent solutions (of approximately the same fitness) have been found and only the best representative is presented. The results obtained

Table 3. Specification for the low-pass Chebychev filter

Frequency (Hz)	$\bar{I}_{loss}$ (dB)
0.02, 0.04, 0.06 . . . 1	< 0.32
1.3	> 52

Table 4. Results of example 2

Component	Evolutionary Strategy $a = 0.5$ $b = 0.5$		Evolutionary Strategy $a = 0.9$ $b = 0.1$		Parametric sampling with 500 point data base		Statistical design (worst-case centered design)		Kjellstrom's design with 800 analysis	
	Δ[%]	tol[%]	Δ[%]	tol[%]	Δ[%]	tol[%]	Δ[%]	tol[%]	Δ[%]	tol[%]
L1	-5.99	8.64	6.3	2.7	0.12	2.49	-5.82	0.62	-0.04	1.62
L2	1.81	3.43	-5.9	3.44	-0.3	1.97	-6.58	0.58	-0.2	1.83
L3	-1.99	4.32	0.02	3.35	0.16	1.84	-6.86	0.45	-0.67	1.08
L4	-15.89	2.81	-9.72	4.43	-0.34	2.23	-6.58	0.58	-0.16	1.62
L5	13.47	2.48	-5.7	5.25	0.66	3	-5.82	0.62	-0.67	1.86
C6	17.39	7.33	18.73	8.05	0.43	3.53	9.35	0.85	0.65	1.41
C7	-0.96	4.17	-2.26	4.06	-0.7	2.05	6.99	0.57	-0.47	1.32
C8	-2.99	5.51	2.31	2.89	1.47	1.8	6.89	0.51	0.9	1.26
C9	19.89	5.34	8.41	4.9	-1.08	2.4	6.89	0.51	-0.11	1.32
C10	-4.03	1.69	5.98	3.73	-1.38	2.34	6.99	0.57	0.58	1.26
C11	3.8	6.75	6.26	9.32	1.09	2.39	9.35	0.85	3.54	3.12
Fitness value	**0.49**		**0.9**		**-**		**-**		**-**	
Yield Y	**92**		**94.33**		**71.33**		**100**		**98.67**	
Cost C_s	**3.14**		**2.83**		**6.76**		**18.68**		**7.48**	

by means of alternative methods [1, 2] are also presented in Tables 2 and 4. In both examples, for the first factor of (4) dominating, increase of the yield has been observed as well as decrease of tolerances. The obtained cost and yield are comparable to those obtained by means of other methods in the first example and explicitly better for Chebychev filter, what proves superiority of the evolutionary approach.

5 Conclusions

New approach to design centering and tolerancing has been proposed. For the presented computational examples, as well as for other examples of the same size (number of parameters), significant enhancement of production yield can be observed. Additionally, it has to be emphasized that evolutionary strategy always gives more than one equivalently good solution, which means greater flexibility of evolutionary approach. Moreover, it does not require costly mapping of feasible region, as in geometrical approach, and does not require huge number of Monte Carlo simulations, as in statistical approach. The maximum total of such simulations is $(\mu + \lambda)It_{max}$, however, it has been observed that solution has been usually found already after few iterations. Thus, it can be concluded that rough estimate of the method order of complexity is comparable with other methods, or even smaller. Furthermore, throw-away cost model has been introduced. The obtained results once more have confirmed efficiency of the presented evolutionary approach.

References

1. Singhal, K., Pinel, J., (1981), "Statistical design centering and tolerancing using parametric sampling" Circuits and Systems, IEEE Transactions, vol. 28, pp. 692-702.
2. Bandler, J.W., Abdel-Malek, H.L., (1977), "Optimal centering, tolerancing and yield determination via updated approximation and cuts" IEEE Trans. Circuits Syst.,vol. CAS-24, pp. 363-372.
3. Elias, N., (1975), "New statistical methods for assigning device tolerances" Proc. 1975 IEEE Int. Symp. Circuits and Syst., pp. 329-332.
4. Harnisch, T., Kunnert, J., Toepfer, H., Uhlmann, F.H., (1997), "Design Centering Methods for Yield Optimization of Cryoelectronic Circuits" IEEE Transactions on Applied Superconductivity, vol. 7, pp. 3434-3437.
5. Abdel-Malek, H.L., Abdel-Karim, S.O., (1991), "The Ellipsoidal Technique for Design Centering and Region Approximation" IEEE Transactions on Computer Aided Design, vol. 10, pp. 1006-1014.
6. Tan, H.K., Ibrahim, Y., (1999), "Design Centering Using Momentum Based CoG" Eng. Opt., vol. 32, pp. 79-100.
7. Sapatnekar, S.S., Vaidya, P.M., Kang, S., (1994), "Convexity-Based Algorithms for Design Centering" IEEE Transactions on Computer Aided Design, vol. 13, pp. 1536-1549.
8. Director, S.W., Hachtel, G.D., (1977), "The simplicial approximation approach to design centering" IEEE Trans. CAS, vol. 24, pp. 363-371.
9. Abdel-Malek, H.L., Bandler, J.W., Biernacki, R.M., (1982), "A one dimensional minimax algorithm based on biquadratic models and its application in circuit design" IEEE Trans. Circuits Syst., vol. CAS-29, pp. 58-62.
10. Spence, R., Soin, R.S., (1988), Tolerance Design of Electronic Circuits, 1st ed., Addison-Wesley Publishers Ltd.
11. Schwencker, R., Schenkel, F., Graeb, H., Antreich, K., (2000), "The generalized boundary curve - A common method for automatic nominal design and design centering of analog circuits" DATE, p. 42.
12. Asselmeyer, T., Ebeling, W., Rose, H., (1997), "Evolutionary Strategies of Optimization" Phys. Rev. E, vol. 56, pp. 1171-1180.
13. Macura, A., Rutkowski, J., (1982), "New method of approximating the acceptable region of nonlinear resistive networks" Electronic Letters, vol. 18, pp. 654-656.
14. Ostermeier, Gawelczyk, Hansen, (1994), "Step-Size Adaptation Based on Non-Local Use of Selection Information" PPSN3, pp. 189-198.
15. Kjellstrom, G., Taxen, L., Blomgren, L.E., (1975), "Optimisation methods for statistical network design" IEEE Int. Symp. Circuits and Syst., pp. 321-324.

Curve Fitting with NURBS using Simulated Annealing

M. Sarfraz and M. Riyazuddin

Information and Computer Science Department, King Fahd University of
Petroleum and Minerals, Dhahran, 31261, Saudi Arabia

Summary. The global optimization strategy of Simulated Annealing is applied to the optimization of knot parameters of NURBS for curve fitting, the objective being the reduction of fitting error to obtain a smooth curve. This is accomplished by using a unit weight vector and a fixed number of control points calculated using the least squares technique, while the sum of squared errors is taken as the objective function.

Keywords: curve fitting, NURBS, approximation, simulated annealing, algorithm

1 Introduction

Researchers in the past have spent considerable time, figuring out how best to fit curves to a set of data points. Curve fitting plays an essential part in many applications. Scientists use curve fitting in application such as data reduction, approximating noisy data, curve and surface fairing and image processing application like generating smooth curves to digitized data [1].

In curve fitting, an input array of data is to be represented mathematically by some analytical function. Since the shape of the underlying function of data is frequently complicated, it is difficult to approximate it by a single polynomial. In this case, a spline and its variants are the most appropriate approximating functions [22].

A *k*th degree B-spline curve is uniquely defined by its control points and knot values, while for NURBS curves, the weight vector has to be specified in addition [3]. The shape modification of these curves plays a central role in CAD and CAGD. Through the manipulation of control points, weights, and/or knot values, users can design a vast variety of shapes using NURBS. Despite NURBS' power and potential, users are faced with the tedium of non-intuitively manipulating a large number of geometric variables [20].

To ameliorate the geometric design with NURBS, a wide variety of techniques for NURBS manipulation have been developed. Typical design techniques include interactive editing, (regular or scattered) data interpolation, shape approximation, cross sectional design, optimization, etc.

Due to the high number of scanned data points, non-deterministic optimization strategies have to be applied to gain optimal approximation results. Evolutionary algorithms show a great flexibility and robustness [10]. Their ability to find solutions even in complex search spaces is one of the reasons for their growing popularity.

Usually subsets of the NURBS parameters are used as independent variables for optimization. The optimization of the control points and then the subsequent knot values was explored in [6] and [17-18]. In [14-15], optimization of both the knots and the weights correspond to the control points for curve fitting. Yau & Chen [21] and then Shalaby et. al. [19] demonstrated that better flexibility of the fitted curve, and hence lower fitting errors, can be obtained by optimizing the control points and then the weights of the NURBS curve.

Genetic algorithms and Tabu search have been applied to optimize NURB parameters. In [14-15], knots and the weights corresponding to the control points have been optimized using genetic algorithms for curve data. In [23], Tabu Search (TS) has been applied to optimize NURBS weights for surface data. In this paper, we have applied Simulated Annealing (SA) to the optimization of NURBS' knots for curve data.

The remainder of the paper is structured as follows. Section 2 surveys the literature related to the optimization of NURBS parameters and application of optimization heuristics to the problem. Section 3 briefly reviews NURBS geometry and its properties. In section 4 we review the simulated annealing optimization heuristic. Section 5 discusses our formulation of the problem to solve it using SA, towards the realization of the NURBS' representation potential. In section 6 we discuss the demonstrated results. Finally, we conclude the paper in section 7.

2 Literature Survey

As NURBS have several control handlers like control points, knot vectors and weights, designers have to decide among them to select the parameters to vary and get the desired shapes. This is one of the most important issues in CAGD.

In [9], approximate estimation of the knot values and optimization of values of points has been made. In [4], simulation of various facial expressions in animation is performed by fixing the control points and changing weights, while in [11], optimization of the knot vector for multi-curve B-spline approximation is performed.

Usually subsets of the NURBS parameters are used as independent variables for optimization. The optimization of control points and subsequently knot values was explored in [17-18] and [6]. In [14-15], optimization of both the knots and the weights corresponding to the control points for curve and surface fitting is done. Better flexibility of the fitted curve, and hence lower fitting errors, can be obtained by optimizing over the control points and then the weights of a NURBS surface [19].

The control points of a B-Spline / NURBS representation of a fitted curve have been traditionally estimated using least squares. The knot values are either taken to be uniform or approximated according to the distribution of the measured points [8] and the weights are set to unity. After the estimation of the control points, the fitting is further enhanced, by optimizing either the knot values or the weights. This enhancement is usually solved as a non-linear programming problem. Gradient-based methods, such as Levenberg-Marquardt method [13], have been used for knot value optimization [17-18]. Direct search methods, such as Powell method, have also been used for the weights and knot optimization [21]. Both the above search methods have the advantage of rapid convergence, but on the other hand may linger in local minima.

In [6], the error minimization of parametric surfaces as a global optimization problem is shown, and it used binary-coded GA's [2] for knot values optimization. But the binary representations of the independent variables tend to enlarge the search space. In [22], a new method that determines the number of knots and their locations simultaneously and automatically by using a genetic algorithm (GA) is discussed. This has the same problem of enlarged searched space. In [14-15], both the knots and the weights corresponding to the control points using GA's have been optimized. The chromosomes have been constructed by considering the candidates of the locations of knots as genes. In [19], real-coded GA's for the optimization of the NURBS weights have been optimized. However, GA's need a large number of objective function evaluations and hence can be used only for fitting small surface patches.

A modified Tabu Search (TS) global optimization technique has been used in [23] to optimize NURBS' weights to minimize the fitting error in surface fitting, but a clear stopping criterion has not been used for this modified Tabu Search algorithm. To our knowledge, the SA global optimization heuristic has not been applied to optimize NURBS parameters in any paper.

3 NURBS

In this section, the formulation of the NURBS curves is first reviewed, followed by their analytic and geometric properties.

A NURBS curve generalizes a B-spline curve. It is the rational combination of a set of piecewise basis functions with n control points p_i and their associated weights $w_{i:}$

$$\mathrm{c}(u) = \frac{\sum_{i=1}^{n} p_i w_i B_{i,k}(u)}{\sum_{i=1}^{n} w_i B_{i,k}(u)}$$

where u is the parametric variable and $B_{i,k}(u)$ is B-spline basis functions. Assuming basis functions of degree k-1, a NURBS curve has $n + k$ knots t_i in non-decreasing sequence : $t_1 \leq t_2 \leq ... \leq t_{n+k-1} \leq t_{n+k}$. The basis functions are defined recursively using uniform knots as

$$B_{i,1}(u) = \begin{cases} 1 \text{ for } t_i \leq u < t_{i+1} \\ 0, \text{ otherwise} \end{cases}$$

with

$$B_{i,k}(u) = \frac{u - t_i}{t_{i+k-1} - t_i} B_{i,k-1}(u) + \frac{t_{i+k} - u}{t_{i+k} - t_{i+1}} B_{i+1,k-1}(u).$$

The parametric domain is $t_k \leq u \leq t_{n+1}$. From users' point of view, the NURBS knots are used to define B-spline basis functions implicitly. Note that, only the relative positions of consecutive knots are actually used to determine its geometry.

NURBS generalize polynomial-based parametric representations for shape modeling. Analogous to B-splines, the rational basis functions of NURBS sum to unity, they are infinitely smooth in the interior of a knot interval provided the denominator is not zero, and at a knot they are at least C^{k-1-r} continuous with knot multiplicity r. They inherit many properties from B-splines, such as the strong convex hull property, variation diminishing property, local support, and invariance under affine geometric transformations. Moreover, NURBS have additional properties. NURBS offer a unified mathematical framework for both implicit and parametric polynomial forms. In principle, they can represent analytic functions such as conics and quadrics precisely, as well as free-from shapes

NURBS include weights as extra degrees of freedom, which influence their local shape. NURBS are attracted toward a control point if the corresponding weight is increased and it is pushed away from a control point if the weight is decreased. If a weight is zero, the corresponding rational basis function is also zero and its control points do not affect the NURBS shape.

4 Simulated Annealing

As its name implies, the Simulated Annealing (SA) exploits analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure (the annealing process) and the search for a minimum in a general system. If a physical system is melted and then cooled slowly, the entire system can be made to produce the most stable (crystalline) arrangement, and not get trapped in a local minimum.

The SA algorithm was first proposed in [7] as a means to find equilibrium configuration of a collection of atoms at a given temperature. Kirkpatrick et al [5] were the first to use the connection between this algorithm and mathematical minimization as the basis of an optimization technique for combinatorial (as well as other) problems.

SA's major advantage over other methods is its ability to avoid being trapped in local minima. The algorithm employs a random search, which not only accepts changes that decrease the objective function E, but also some changes that would increase it. The latter are accepted with a probability

$$\text{Prob(accept)} = \exp(-\Delta E/T)$$

where ΔE is the increase in E and T is a control parameter, which by analogy with the original application is known as the system "temperature" irrespective of the objective function involved.

Briefly SA works in the following way. Given a function to optimize, and some initial values for the variables, simulated annealing starts at a high, artificial, temperature. While cooling the temperature slowly, it repeatedly chooses a subset of the variables, and changes them randomly in a certain neighborhood of the current point. If the objective function has a lower function value at the new iterate, the new values are chosen to be the initial values for the next iteration. If the objective function has a higher function value at the new iterate, then the new values are chosen to be the initial values for the next iteration with a certain probability, depending on the change in the value of the objective function and the temperature.

The higher the temperature and the lower the change, the more probable that the new values are chosen to be the initial variables for the next iteration. Throughout this process, the temperature is decreased gradually, until eventually the values do not change anymore. Then, the function is presumably at its global minimum. The global minimum is obtained by choosing an appropriate "cooling schedule" which includes the temperature and its cooling rate. A cooling schedule describes the temperature parameter T, and gives rules for lowering it as the search progresses. The simulated annealing algorithm, as in [16], is shown in Figure 1.

```
Algorithm Simulated_annealing ($S_0$, $T_0$, $\alpha$, $\beta$, $M$, $Maxtime$);
        (*$S_0$ is the initial solution *)
        (*BestS is the best solution*)
        (*$T_0$ is the initial temperature*)
        (*$\alpha$ is the cooling rate*)
        (*$\beta$ a constant*)
        (*$Maxtime$ is the total allowed time for the annealing process*)
        (*$M$ represents the time until the next parameter update*)
Begin
        $T = T_0$;
        $CurS = S_0$
        $BestS = CurS$;             /*BestS is the best solution seen so far */
        $CurCost = Cost(CurS)$;
        $BestCost = Cost(BestS)$;
        $Time = 0$;
```

```
            Repeat
                Call Metropolis(CurS, CurCost, BestS, BestCost, T,M );
                Time = Time + M;
                T = $\alpha$ T;
                M = $\beta$ M;
            Until (Time $\geq$ MaxTime);
            Return (BestS)
End (* of Simulated Annealing *)

Algorithm Metropolis(CurS, CurCost, BestS, BestCost, T, M );
Begin
    Repeat
        NewS = Neighbor(CurS);
        NewCost = Cost(NewS);
        $\Delta$Cost = (NewCost – CurCost);
        If ($\Delta$Cost < 0) Then
            CurS = NewS;
                If NewCost < BestCost Then
                    BestS = NewS
                EndIf
        Else
            If (RANDOM < $e^{-\Delta Cost/T}$) Then
                CurS = NewS;
            EndIf
        EndIf
        M = M – 1
    Until (M = 0)
End (*of Metropolis*)
```

Figure 1. Simulated Annealing algorithm as in [16].

The core of the algorithm is the *Metropolis* procedure, which simulates the annealing process at a given temperature *T*. The Metropolis procedure receives as input the current temperature *T*, and the current solution *CurS*, which it improves through local search. *Metropolis* must also be provided with the value *M*, which is the amount of time for which annealing must be applied for a temperature *T*.

The procedure Simulated_annealing simply invokes *Metropolis* at decreasing temperatures. Temperature is initialized to a value T_0 at the beginning of the procedure, and is reduced in a controlled manner (typically in a geometric progression); the parameter α is used to achieve this cooling. The amount of time spent in annealing at a temperature is gradually increased as temperature is lowered. This is done using the parameter β (>1). The variable *Time* keeps track of the time being expended in each call to the *Metropolis*. The annealing procedure halts when *Time* exceeds the allowed time.

The *Metropolis* procedure uses the procedure *Neighbor* to generate a local neighbor *NewS* of any given solution *S*. The function *Cost* returns the cost of a given solution *S*. If the cost of the new solution *NewS* is better than the cost of the current solution *CurS*, then the new solution *NewS* is accepted, and we do so by setting *CurS* equal to *NewS*. If the cost of the new solution *NewS* is better than the best solution (*BestS*) seen thus far, then we also replace *BestS* by *NewS*. If the new solution has a higher cost in comparison to the original solution *CurS*, *Metropolis* will accept the new solution on a probabilistic basis. A random number (*RANDOM*) is generated in the range 0 to 1. If this random number is smaller than $e\text{-}\Delta Cost/T$, where $\Delta Cost$ is the difference in costs, and T is the current temperature, then the uphill solution is accepted. This criterion for accepting the new solution is known as the Metropolis criterion. The *Metropolis* procedure generates and examines M solutions.

The probability that an inferior solution is accepted by the *Metropolis*, is given by

$$P(RANDOM < e\text{-}\Delta Cost/T).$$

The random number generation is assumed to follow a uniform distribution. Remember that $\Delta Cost > 0$ since we have assumed that *NewS* is uphill from *CurS*. At very high temperatures, (when $T \to \infty$),

$$e\text{-}\Delta Cost/T \approx 1.$$

Hence the above probability approaches 1. On the contrary, when $T \to 0$, the probability $e\text{-}\Delta Cost/T$ falls to 0.

In order to implement simulated annealing, we need to formulate a suitable cost function for the problem being solved. In addition, as in the case of local search techniques, we assume the existence of a neighborhood structure, and need *Neighbor* function to generate new states (neighborhood states) from current states. And finally we need a cooling schedule that describes the temperature parameter T and gives rules for lowering it.

5 The Proposed Method

Figure 2 shows the basic building blocks of our implemented system. A digitized image is obtained from an electronic device or by scanning an image. The quality of digitized scanned image depends of various factors such as the image on paper, scanner type and the attributes set during scanning. The quality of digitized image obtained directly from an electronic device depends on the resolution device, source of image, type of image, etc.

The contour of the digitized image is extracted using the boundary detection algorithms. There are numerous algorithms for detecting boundary. We used the algorithm proposed by Quddus [12]. The input to this algorithm is a bitmap file. The algorithm returns a number of segments and for each segment, a number of boundary points and their values.

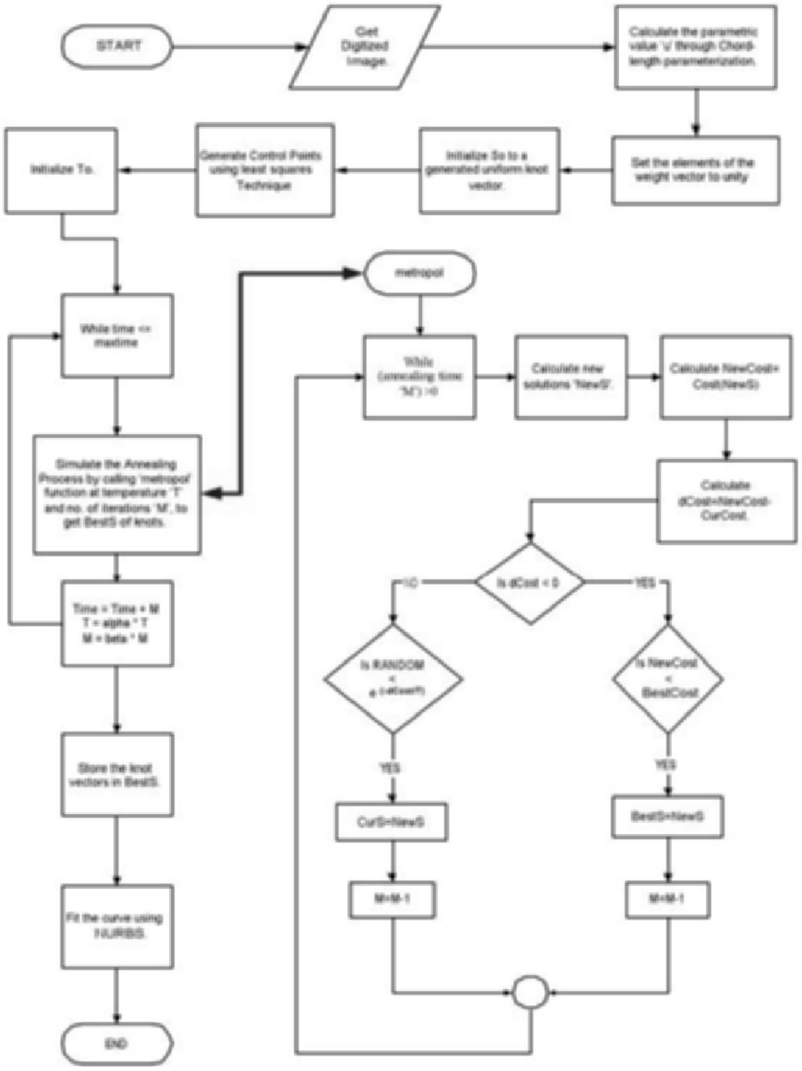

START
Get Digitized Image.
Calculate the parametric value 'u' through Chord-length parameterization.
Set the elements of the weight vector to unity
Initialize So to a generated uniform knot vector.
Generate Control Points using least squares Technique
Initialize To.
While time <= maxtime
Simulate the Annealing Process by calling 'metropol' function at temperature 'T' and no. of iterations 'M', to get BestS of knots.
Time = Time + M
T = alpha * T
M = beta * M
Store the knot vectors in BestS.
Fit the curve using NURBS.
END
metropol
While (annealing time 'M') >0
Calculate new solutions 'NewS'.
Calculate NewCost= Cost(NewS)
Calculate dCost=NewCost-CurCost.
Is dCost < 0
NO
YES
Is RANDOM < e (-dCost/T)
Is NewCost < BestCost
YES
YES
CurS=NewS
BestS=NewS
M=M-1
M=M-1

Figure 2. The basic building blocks of our implemented system

The parameter value t_j for each data point is a measure of the distance of the data point along the curve. One useful approximation for this parameter value uses the chord length between data points. Specifically, for j data points, the parameter value at the ℓ^{th} data point is

$$t_1 = 0$$

$$\frac{t_\ell}{t_{\max}} = \frac{\sum_{s=2}^{\ell} |D_s - D_{s-1}|}{\sum_{s=2}^{j} |D_s - D_{s-1}|} \qquad \ell \geq 2$$

The maximum parameter value, $t_{\max}$, is usually taken as the maximum value of the knot vector. The control points are calculated using the least squares technique. A fairer or smoother curve is obtained by specifying fewer control polygon points than data points, i.e. $2 \leq k \leq n < j$. Recalling that a matrix times its transpose is always square, the control polygon for a curve that fairs or smoothes the data is given by

$$[D] = [B]\,[P]$$
$$[B]^{\mathrm{T}}[D] = [B]^{\mathrm{T}}[B]\,[P]$$
$$[P] = [[B]^{\mathrm{T}}[B]]^{-1}[B]^{\mathrm{T}}[D]$$

where $[D]^{\mathrm{T}} = [\ D_1(t_1)\ D_2(t_2)\ .\ .\ .\ D_j(t_j)\ ]$ are data points, $[P]^{\mathrm{T}} = [\ P_1\ P_2\ .\ .\ .\ P_{n+1}\ ]$ are the control points and [B] is the set of B-spline basis functions.

The evaluation of the control points by least squares approximation can be viewed as an initial estimation of the fitted curve. Further refinement can be obtained by optimizing the different NURBS parameters, such as the knot values and the weights in order to achieve better fitting accuracy. The error function (or cost function or the *objective function*) between the measured points and the fitted curve is generally given by the following equation:

$$E = \left(\sum_{i=0}^{s} |Q_i - S(\alpha_1, ..., \alpha_n)|^r / s \right)^{1/r},$$

where Q represents the set of measured points; $S(\alpha_1, \ldots, \alpha_n)$ is the geometric model of the fitted curve, where $(\alpha_1, \ldots, \alpha_n)$ are the parameters of the fitted curve; s is the number of measured points and r is an exponent, ranging from 1 to infinity. The fitting task can then be viewed as the optimization of the curve parameters $(\alpha_1, \ldots, \alpha_n)$ to minimize the error (or cost) E. In case the exponent r is equal to 2, the above equation reduces to the least squares function.

We have used the Simulated Annealing optimization heuristic to optimize knots of the NURBS curve. Figure 1 shows the algorithm used. The initial solution S_0 is a uniform knot vector, with a range of [0, npts+k-1].

The cooling schedule used here is presented in [5]. It is based on the idea that the initial temperature T_0 must be large to virtually accept all transitions and that the changes in the temperature at each invocation of the Metropolis loop are small. The scheme provides guidelines to the choice of T_0, the rate of decrements of T, the termination criterion and the length of the markov chain (M).

Initial Temperature To: The initial temperature must be chosen so that almost all transitions are accepted initially. That is, the initial acceptance ratio $\chi(T_0)$ must be close to unity where

$\chi(T_0)$=(Number of moves accepted at T_0)/(Total number of moves attempted at T_0)

To determine T_0, we start off with a small value of initial temperature given by T'_0, in the metropol function. Then $\chi(T'_0)$ is computed. If $\chi(T'_0)$ is not close to unity, then T'_0 is increased by multiplying it by a constant factor larger than one. The above procedure is repeated until the value of $\chi(T'_0)$ approaches unity. The value of T'_0 is then the required value of T_0.

Decrement of T: A decrement function is used to reduce the temperature in a geometric progression, and is given by

$$T_{k+1} = \alpha\, T_k\,, \qquad k = 0,1,\, \ldots\,,$$

where α is a positive constant less than one, since successive temperatures are decreasing. Further, since small changes are desired, the value of α is chosen very close to unity, typically $0.8 \leq \alpha \leq 0.99$.

Length of Markov chain M: This is equivalent to the number of times the Metropolis loop is executed at a given temperature.

If the optimization process begins with a high value of To, the distribution of relative frequencies of states will be very close to the stationary distribution. In such a case, the process is said to be in quasi equilibrium. The number M is based on the requirement that at each value of T_k quasi equilibrium is restored.

Since at decreasing temperatures uphill transitions are accepted with decreasing probabilities, one has to increase the number of iterations of the Metropolis loop with decreasing T (so that the Markov chain at that particular temperature will remain irreducible and with all states being non null). A factor β is used ($\beta > 1$) which, in a geometric progression, increases the value of M. That is, each time the Metropolis loop is called, T is reduced to $\alpha\mathrm{T}$ and M is increased to βM.

The neighborhood of the current solution '*CurS*' is generated in the neighborhood of [*CurS* - 0.001, *CurS* + 0.001].

6 Experimental Results

We used the ‘pound’ image symbol as the input image for our algorithm, as shown in Figure 3(a). Table 1 explains various information for the selection of concerned parameters. The number of control points, in the ‘pound’ image symbol , have been taken as 70. While cooling, since small changes in temperatures are desired, we have chosen the value of α as 0.99, which is close to unity. Since the value of β should be greater than 1, a value of 1.5 is chosen. The algorithm executes the *Metropol* function, based on *Maxtime*, which is set to 250. Since we want to deal with cubic curves, the order *K*, of the curves is chosen to be 4.

Table 1. Selection of parameters in the experimentation.

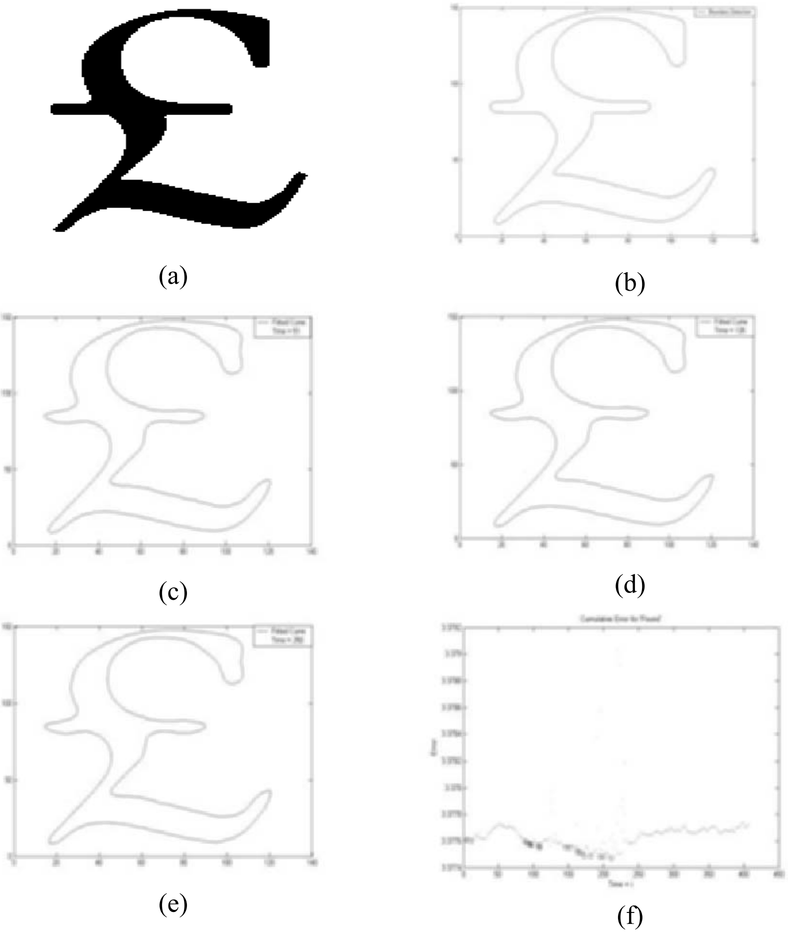
Parameter	Value
Number of control points	70
M	50
α	0.99
β	1.5
Maxtime	250
K (order)	4

Figure 3(a) shows the original scanned image given as an input to the algorithm. Figure 3(b) shows the outline of the image obtained after applying the boundary detection algorithm. Figures 3(c) & 3(d) depict the intermediate fittings of the ‘pound’ symbol at iterations (Time + *i*) = 51 & 126. Figure 3(e) shows the fitting for the actual iteration of 250 (Maxtime), where ‘*i*’ iterates over Annealing time ‘M’. Figure 3(f) depicts the actual reduction in the cost (error) as the number of iterations increase, where a gradual decrease in the (current) cost function can be viewed. The figure also shows that (current) costs are selected for the next iteration, even if previous (current) costs were better, to avoid getting trapped in the local minimum. Finally the best cost is selected after the algorithm completes the specified number of iterations. Table 2 shows the actual number of times that the Metropol function is executed.

Table 2. Execution of the Metropol function.

S.No	*Time=Time+M*	$M=\beta*M$
1	1	50
2	51	75
3	126	112.5
4	238.5	168.75

Table 2 shows that, the Metropol function executes Time + *M* i.e. 238.5 + 168.75, which is equal to 407 number of times, which is correctly shown in figure 3(f). Figure 3(f) also shows the best cost (least error) for the ‘pound ‘ symbol to be 3.378, while the execution time is 530.859 seconds, which is very less compared to the (nearly 10 to 12) hours taken for a similar kind of optimization by [14-15] using genetic algorithms.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 3. Implementation of the algorithm for the “pound” image.

7 Conclusions

The objective of the research presented in this paper was to develop an algorithm for the global optimization of the fitting error between a set of scanned points and a fitted curve. To achieve this objective, the Simulated Annealing optimization heuristic was tailored to solve the problem. From our work, we conclude that Simulated Annealing optimization heuristic when used for solving, the problem at hand, gives better results. The SA algorithm uses an efficient local optimization method, which ensures it's accurate arrival at the global optimum. One of our immediate concerns is to find appropriate metrics to compare our results with the results of [23] who used Tabu Search for a similar problem. We would also like to compare our results with other global optimization heuristics.

In future, this work can be extended to optimize two or more NURBS parameters like control point-weight, knot vector-weight, etc. It can also be applied to NURBS surfaces, instead of NURBS curves. Other appropriate global optimization techniques can be applied to optimize NURBS parameters to solve the problem. Also, this work can be incorporated in the reverse engineering component of the CAD/CAM modeling software.

Acknowledgement

This work has been supported by King Fahd University of Petroleum and Minerals under the funded project entitled "Reverse Engineering for Geometric Models Using Evolutionary Heuristics.

References

1. Chou J.J. & Piegl L.A. (1992), "Data Reduction Using Cubic Rational B-Splines," IEEE Computer Graphics & Applications.
2. Goldberg D.E. (1989), "Genetic Algorithms in Search," Optimization and Machine Learning, Addisson-Wesley.
3. Hoffmann M. & Juhasz I (2001), "Shape Control of Cubic B-spline and NURBS Curves by Knot Modifications", Proceedings of the fifth international conference on information visualizaton (IV'2001)-UK, IEEE Computer Society.
4. Huang D. & Yan H. (2003), "NURBS Curve Controlled Modelling for Facial Animation," 27, Computers & Graphics, pp. 373-385.
5. Kirkpatrick S., Gelatt Jr. C. & Vecchi M. (1983), "Optimization by Simulated Annealing,", Science, 220(4598): pp. 498-516.
6. Limaiem A., Nassef A. & Elmaghraby H.A. (1996), "Data Fitting using Dual Krigging and Genetic Algorithms," CIRP Annals, Vol. 45, pp. 129-134.
7. Metropolis N., Roshenbluth A., Rosenbluth M., Teller A. & Teller E. (1953), "Equation of State Calculations by Fast Computing machines," J. Chem. Phys., Vol.21, No. 6, pp. 1087-1092.
8. Piegl L. (1991), "On NURBS: A Survey," IEEE computer graphics & applications, Vol. 11(1): pp. 55-71.
9. Piegl L., & Tiller W. (1995), "The NURBS Book," Springer-Verlag, Berlin.
10. Pontrandolfo F., Monno G. & Uva A.E. (2001), "Simulated Annealing Vs

Genetic Algorithms for Linear Spline Approximation of 2D Scattered Data," Proc. of XII adm international conference, International Conference on Design Tools and Methods in Industrial Engineering, Rimini, Italy.

11. Prahasto T. & Bedi S. (2000), "Optimization of Knots for the Multi Curve B-Spline Approximation," IEEE conference on geometric modeling and processing, Hong Kong, China.
12. Quddus A. (1998), "Curvature Analysis Using Multi-resolution Techniques,". PhD Thesis, King Fahd University of Petroleum & Minerals. Dhahran, Saudi Arabia.
13. Rao S. S. (1999), "Engineering Optimization, Theory and Practice," John-Wiley and Sons, New York.
14. Sarfraz, M, and Raza, A, (2002), Visualization of Data using Genetic Algorithm*, Soft Computing and Industry: Recent Applications*, Eds.: R. Roy, M. Koppen, S. Ovaska, T. Furuhashi, and F. Hoffmann, ISBN: 1-85233-539-4, Springer, 535 - 544.
15. Sarfraz, M., and Raza, S. A., (2001), Capturing Outline of Fonts using Genetic Algorithm and Splines, The Proceedings of IEEE International Conference on Information Visualization-IV'2001-UK, IEEE Computer Society Press, USA, 738-743.
16. Sait, S. M. and Youssef, H. (1999), Iterative Computer Algorithms with Applications in Engineering: Solving Combinatorial Optimization Problems, 1st edition, IEEE Computer Society Press Los Alamitos, CA, USA.
17. Sarkar B. and Menq C. H. (1991), "Smooth Surface Approximation and Reverse Engineering," Computer Aided Design, Vol. 23, pp. 623-628.
18. Sarkar B. & Menq C. H. (1991), "Parameter Optimization in Approximating Curves and Surfaces to Measurment Data," Computer Aided Geometric Design, Vol. 8, pp.267-290.
19. Shalaby M. M., Nassef A. O., and Metwalli S. M., (2001), "On the Classification of Fitting Problems for Single Patch Free-Form Surfaces in Reverse Engineering," Proceedings of the ASME Design Automation Conference, Pittsburgh.
20. Xie H. & Qin H. (2001), "Automatic Knot Determination of NURBS for Interactive Geometric Design," IEEE conference on Shape Modelling & Applications, Genova , Italy.
21. Yau H. T., and Chen J. S. (1997), "Reverse Engineering of Complex Geometry Using Rational B-Splines," International Journal of Advanced Manufacturing Technology, Vol. 13, pp. 548-555.
22. Yoshimoto Y., Moriyama M. and Harada T. (1999), "Automatic Knot Replacement by a Genetic Algorithm for Data Fitting with a Spline," Proceedings of the International Conference on Shape Modeling and Applications, Aizu-Wakamatsu, Japan, pp.162–169.
23. Youssef A. M. (2001), "Reverse Engineering of Geometric Surfaces using Tabu Search Optimization Technique," Master Thesis, Cairo University, Egypt.

Multiobjective Adaptive Representation Evolutionary Algorithm (MAREA) - a new evolutionary algorithm for multiobjective optimization

Crina Groşan

Department of Computer Science
Babeş-Bolyai University, Kogălniceanu 1
Cluj-Napoca, 3400, Romania.
cgrosan@cs.ubbcluj.ro

Summary. Many algorithms for multiobjective optimization have been proposed in the last years. In the recent past a great importance have the MOEAs able to solve problems with more than two objectives and with a large number of decision vectors (space dimensions). The difficulties occur when problems with more than three objectives (higher dimensional problems) are considered. In this paper, a new algorithm for multiobjective optimization called Multiobjective Adaptive Representation Evolutionary Algorithm (MAREA) is proposed. MAREA combines an evolution strategy and an steady-state algorithm. The performance of the MAREA algorithm is assessed by using several well-known test functions having more than two objectives. MAREA is compared with the best present day algorithms: SPEA2, PESA and NSGA II. Results show that MAREA has a very good convergence.

1 Introduction

In the recent years a number of Multiobjective Optimization Evolutionary Algorithms (MOEAs) have been proposed. The interest is now focused on finding the Pareto front for functions having more than two objectives.

A new algorithm called Multiobjective Adaptive Representation Evolutionary Algorithm (MAREA) is proposed in this paper. This algorithm uses a solution representation similar to the representation used by Adaptive Representation Evolutionary Algorithm (AREA) introduced in [5]. Used operators are the same as those used by AREA.

For assessing the performances of the MAREA algorithm a comparison with some recent algorithms for Multiobjective Optimization is performed. The algorithms used in this comparison are: SPEA2 introduced by Zitzler and al. in [8], NSGA II introduced by Deb in [2] and PESA introduced by

Knowles in [1]. Three difficult test functions are considered for comparison purposes. The number of objectives varies between 2 and 8.

The paper is structured as follows: Section 2 describes AREA technique: solution representation and the operators used. In Section 3 the newly algorithm proposed – MAREA – is presented. In Section 4 the test function used for comparison are described. Two performance metrics for compare the results obtained by these algorithms are presented in Section 5. Numerical experiments with the algorithms SPEA2, NSGA II and PESA are performed in Section 6 of the paper. A set of conclusions are mark out in Section 7 of the paper.

2 AREA Technique

The main idea of this technique is to allow each solution to be encoded over a different alphabet. Moreover, the representation of a particular solution is not fixed. Representation is adaptive and may be changed during the search process as effect of the mutation operator. AREA relies mainly on Dynamic Representation (DR) proposed in [7].

2.1 Solution representation

Within AREA each solution has its own encoding alphabet and can be represented as a pair(x, B) where B is an integer number, $B \geq 2$ and x is a string of symbols from the alphabet $\{0, 1, \ldots, B\text{-}1\}$. If B= 2, the standard binary encoding is obtained. The alphabet over which x is encoded may change during the search process by applying an operator similar with mutation operator and called *transmutation*. If no ambiguity arises we will use B to denote the alphabet $B = \{0, 1, \ldots B\text{-}1\}$.

2.2 Mutation

Each gene is affected by mutation with a fixed mutation probability. The mutated gene is a symbol randomly chosen from the same alphabet.

Consider the chromosome C represented over the alphabet B = 8: C = (631751, 8). Consider a mutation occurs on the 3^{rd} position of x. Let be 4 the value of mutated gene. Then the mutated chromosome is: C_1 = (634751, 8).

2.3 Transmutation

The transmutation operator can modify only the value of the alphabet over that the chromosome is represented. The new value of the alphabet is a randomly chosen integer value.

When position giving the alphabet B is changed, then the object variables will be represented using symbols over the new alphabet, corresponding to the mutated value of B.

Consider a transmutation occurs on the last position of chromosome C and the mutated value is $B_2 = 10$. Then the mutated chromosome is:

$C_2 = (209897, 10)$.

C and C_2 encode the same value over two different alphabets ($B = 8$, $B_2 = 10$).

Remark A mutation generating an offspring worse than its parent is called a *harmful mutation*.

2.4 Offspring acceptance

If the offspring obtained by mutation is better than its parent than the parent is removed from the population and the offspring enters the new population. Otherwise, a new mutation of the parent is considered.

A mutation generating an offspring worse than its parent is called a harmful mutation. If the number of successive harmful mutations exceeds a prescribed threshold (*Max_Harmful_Mutations*) the individual representation is changed and with this new representation it enters in the new population.

3 MAREA Algorithm

Two stages are distinguished to determinate the real Pareto front. At the first stage the effort is focused on finding the Pareto front. At the second stage the effort is focused on spreading the individuals along the Pareto front.

At stage I, MAREA uses a population containing few individuals (usually one individual). These individuals are mutated until no improvement of solution's quality occurs for a specified number of iterations. That means that this solution is near Pareto front (the real Pareto front or a local Pareto front). Stage II begins at this moment. At this stage the spread of solution on the Pareto front is realized. The aim of this stage is to obtain a good distribution along Pareto front. An enlarged population is permitted at this stage in order to obtain a good coverage of Pareto front. A detailed description of the MAREA's stages is given bellow.

3.1 Stage I – Convergence to the Pareto front

MAREA uses a single population of individuals that are evolved using two variations operators: mutation and transmutation. The evolutionary model used here is similar to (1+1) ES.

At this stage the population has only one individual. Initial population (initial individual) is randomly generated. Mutation is the unique variation operator. The offspring and parent are compared. Dominance relation guides the survival.

If the offspring dominates the parent then the offspring replaces the parent. If the parent dominates the offspring obtained for Max_Harmful_Mutations successive mutations then another alphabet is chosen and the parent is represented in symbols over this alphabet.

A parameter called *Max_Steps_To_Front* is used on order to find when this stage is over. If no improvement occurs for *Max_Steps_To_Front* iterations it is likely that this solution is near Pareto (global or very good local) front. As the search process gets near Pareto front the first stage stops here and at this moment the second stage starts.

3.2 Stage II – Dispersion on the Pareto front.

After first stage the population contains one individual only. This individual is near Pareto (local or global) front. The evolutionary model used here is similar with Steady-State.

During this stage the population size is variable and can increase and decrease also. Mutation operator is applied to each solution from population. *Max_Harmful_Mutation* parameter is used to determinate when the alphabet representation for a solution is changed. At each step a randomly chosen individual is mutated. The offspring is added to the population. All dominated solutions are removed. If all solutions are nondominated and the population size exceeds a prescribed threshold then some solutions are removed. For computing which solutions will be removed from the population the following mechanism is used: the nearest solutions are chosen from population. One of them is removed.

The size of final population can be smaller than maximum size allowed. This situation is likely if the Pareto front consist of a very small number of points.

4 Test Functions

Test functions used in these experiments were introduced in [3]. Each of them is described in below. These test functions are M - objective problems.

Test function DTLZ1

Test function DTLZ1 is defined as follows:

$$\begin{aligned}
&f_1(\mathrm{x}) = \tfrac{1}{2}x_1x_2\ldots x_{M-1}(1+g(\mathrm{x}_M)),\\
&f_2(\mathrm{x}) = \tfrac{1}{2}x_1x_2\ldots(1-x_{M-1})(1+g(\mathrm{x}_M)),\\
&\vdots\\
&f_{M-1}(\mathrm{x}) = \tfrac{1}{2}x_1(1-x_2)(1+g(\mathrm{x}_M)),\\
&f_M(\mathrm{x}) = \tfrac{1}{2}(1-x_1)(1+g(\mathrm{x}_M)),\\
&\textit{where}\\
&g(\mathrm{x_M}) = 100\cdot\Big(|\mathrm{x}_M| + \textstyle\sum_{x_i\in \mathrm{x_M}}(x_i-0.5)^2 - \cos(20\pi(x_i-0.5))\Big)
\end{aligned}$$

The domain of definition is [0, 1]. The global minimum is ${x_i}^* = 0.5$, (${x_i}^*\in$ x_M) and $\sum\limits_{m=1}^{M} f_m^* = 0.5$.

Test function DTLZ2
Test function DTLZ2 is the following:

$$\begin{aligned}
&f_1(\mathrm{x}) = (1+g(\mathrm{x}_M))\cos(x_1\pi/2)\ldots\cos(x_{M-1}\pi/2),\\
&f_1(\mathrm{x}) = (1+g(\mathrm{x}_M))\cos(x_1\pi/2)\ldots\sin(x_{M-1}\pi/2),\\
&f_M(\mathrm{x}) = (1+g(\mathrm{x}_M))\sin(x_1\pi/2),\\
&\textit{where}\\
&g(\mathrm{x}_M) = \sum_{x_i\in \mathrm{X}_M}(x_i{-}0.5)^2.
\end{aligned}$$

The domain of definition is [0, 1]. The global optimum point is ${x_i}^* = 0.5$, (${x_i}^*\in \mathrm{x}_M$) and $\sum\limits_{m=1}^{M}(f_m^*)^2 = 1$.

Test function DTLZ3
The test functions DTLZ3 are the same that test functions used in DTLZ2 and g function is one used in DTLZ1.
The domain of definition is [0, 1]. The global optimum point is ${x_i}^* = 0.5$, (${x_i}^*\in \mathrm{x}_M$) and

$$\sum_{m=1}^{M}(f_m^*)^2 = 1.$$

5 Performance Metrics

In this study a convergence metrics proposed by Deb and Jain in [4] is used to determinate the algorithms performances.

5.1 Convergence metric

This convergence metric represents the distance between the set of converged non-dominated solutions and the global *PO* front.

Let P^* be the reference or target set of points on the PO front and let F be the final non-dominated set obtained by a multiobjective evolutionary algorithm.

Then from each point i in F the smallest normalized Euclidian distance to P^* will be:

$$d_i = \min_{j=1}^{|P*|} \sqrt{\sum_{k=1}^{M} \left(\frac{f_k(i) - f_k(j)}{f_k^{\max} - f_k^{\min}} \right)^2}$$

where $f_k^{\max}$ and $f_k^{\min}$ are the maximum and minimum function values of k-th objective function in P^* and M represent the number of objective functions.

Remark

Lower values of convergence metric represent good convergence ability.

6 Numerical Experiments

In order to mark out MAREA performances some numerical experiment are performed. Three algorithms are used for comparisons: SPEA2, NSGA II and PESA.

To make a comparison fair the same number of function evaluations is used by all considered algorithms.

The parameters used by each of the considered algorithms are presented in the Table 1.

The values of parameters used by PESA, SPEA2 and NSGA II have taken from [6].

The values corresponding to population size for each considered objectives numbers are presented in Table 2.

The number of generations and the number of functions evaluations corresponding to objectives number for each test functions are presented in Tables 3 and 4.

The results obtained by applying convergence metric are presented in Table 5. Results are averaged over 30 runs. Resutls obtained by NSGA II, SPEA2 and PESA are taken from [6].

We can see from this table MAREA outperforms the considered algorithms for comparison in almost all situations. PESA is better than NSGA II and SPEA 2. MAREA strongly outperforms NSGA II and SPEA 2 for all considered test functions. For a small number of objectives PESA and MAREA give closed results. But for a greater number of objectives MAREA is able to converge to the global Pareto front while PESA not in all situations.

7 Conclusions and Further Work

In this paper a new evolutionary algorithm for multiobjective optimization was proposed. This algorithm uses solution representation introduced by

Table 1. The values of parameters used by MAREA, SPEA2, NSGA II and PESA. The parameters for SPEA2, NSGA II and PESA are taken from [6].

Parameter	**MAREA**	**PESA**	**SPEA2**	**NSGA II**
Crossover probability	-	0.8	0.7	0.7
Distribution Index (DI) for SBX	-	15	15	15
Mutation Probability (if n = number of variables)	$1/n$	$1/n$	$1/n$	$1/n$
DI for polynomial mutation	-	15	15	20
Ratio of internal population size to archive size	-	1:1	1:1	1:1
Number of grids per dimension (PESA)	-	10	-	-
MAX_HARMFUL_ MUTATIONS (MAREA)	3000	-	-	-

Table 2. Population size corresponding to objectives number. The values are taken from [6].

Number of objectives	Population size
2	20
3	50
4	100
6	250
8	400

Table 3. Number of generations used in experiments for each test functions. The values are taken from [6].

Test function	**Number of generations**	
	For 2, 3 and 4 objectives	**For 6 and 8 objectives**
DTLZ1	300	600
DTLZ2	300	600
DTLZ3	500	1,000

Table 4. Number of functions evaluations used for all considered algorithms. The values are taken from [6].

Test function	**Number of functions evaluations**				
	2 obj	**3 obj**	**4 obj**	**6 obj**	**8 obj**
DTLZ1	6,000	15,000	30,000	150,000	360,000
DTLZ2	6,000	15,000	30,000	150,000	360,000
DTLZ3	10,000	25,000	50,000	250,000	600,000

Table 5. Results obtained by applying convergence metric. Results are averaged over 30 runs.

Test function		**Convergence metric**			
		MAREA	**PESA**	**SPEA2**	**NSGA II**
DTLZ1	2 objectives	0.57292	2.8694	3.088	2.2766
	3 objectives	0.04299	0.0441	0.0484	0.3836
	4 objectives	0.01546	0.0231	0.2992	3.1028
	6 objectives	0.001604	0.0017	5.999	120.191
	8 objectives	0.00391	0.004	498.27	465.301
DTLZ2	2 objectives	0.000091	0.00008	0.00026	0.00180
	3 objectives	0.00014	0.00035	0.00663	0.010003
	4 objectives	0.001304	0.00170	0.03369	0.04529
	6 objectives	0.00195	0.00301	2.00216	1.67564
	8 objectives	0.008084	0.00689	2.35258	2.30766
DTLZ3	2 objectives	0.12605	22.52023	16.87313	21.32032
	3 objectives	0.1591	1.80296	2.39884	5.65577
	4 objectives	0.0293	1.16736	4.00596	66.94049
	6 objectives	0.1822	0.15035	217.95360	1273.30601
	8 objectives	0.1734	7.23062	1929.94832	1753.41364

AREA technique. The particularity of this algorithm is his structure. Initially only one individual is considered. The algorithm tries to situate this individual near Pareto front. Mutation and transmutation operators are used in order to improve the individual. If no improvement appear after a specified number of mutations means that this solution in near Pareto front. (Sometimes this can be a local Pareto front). When this unique solution is near Pareto front the population can be extended by considering the others nondominates solutions. These solutions are obtained by applying mutation and transmutation operator also over these new solutions considered. After a specified number of iteration the last population obtained is considered the final population.

The two parameters for determining when the alphabet over which a solution is represented is changed and the other parameter determining when the first stage is finish (i.e. determining when population size can increase) are very important. By considering different values of these parameters the final result is different.

An adaptation of these parameters for each problem is the work of the other paper.

References

1. Corne, D., Knowles, J., Oates, M. The Pareto-Envelope based Selection Algorithm for Multiobjective Optimization. In Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature, Springer-Verlag, Berlin, (2000) 839-848.
2. K. Deb, S. Agrawal, A Pratap and T. Meyarivan, A fast elitist non – dominated sorting genetic algorithm for multi-objective optimization: NSGA II. In M. S. et al. (Ed), *Parallel Problem Solving From Nature – PPSN VI*, Springer-Verlag, Berlin (2000) 849-858.
3. K. Deb, L. Thiele, M. Laumanns and E. Zitzler. Scalable Multi-Objective Optimization Test Problems. Proceeding of IEEE Congress on Evolutionary Computation, Hawaii, (2002).
4. Deb, S. Jain, Running performance metrics for evolutionary multi-objective optimization, KanGAL Report 2002004, Indian Institute of Technology, Kanpur, India, (2002).
5. Grosan, C., Oltean, M. Adaptive Representation Evolutionary Algorithm - a new technique for single objective optimization. In Proceedings of First Balcanic Conference on Informatics (BCI), Thessaloniki, Greece, (2003) 345-355.
6. V. Khare, X. Yao, K. Deb. Performance Scaling on Multi-objective Evolutionary Algorithms. Technical Report 2002009, Kanpur Genetic Algorithm Laboratory (KanGAL), Indian Institute of Technology Kanpur, India (2002).
7. 12.Kingdon J, Dekker L. The Shape of Space, Proceedings of the First IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA '95) IEE, London, (1995) 543-548.
8. Zitzler, E., Marco Laumanns and Thiele, L., SPEA 2: Improving the Strength Pareto Evolutionary Algorithm, TIK Report 103, Computer Engineering and Networks Laboratory (TIK), Departament of Electrical Engineering Swiss federal Institute of Technology (ETH) Zurich, (2001).

Adapting Multi-Objective Meta-Heuristics for Graph Partitioning

R. Baños[1], C. Gil[1], M.G. Montoya[1], and J. Ortega[2]

[1] Dept. Arquitectura de Computadores y Electrónica, Universidad de Almería,
La Cañada de San Urbano s/n, 04120 Almería (Spain)
(rbanos,cgil,mari)@ace.ual.es

[2] Dept. Arquitectura y Tecnología de Computadores, Universidad de Granada,
Periodista Daniel Saucedo s/n, 18071 Granada (Spain) julio@atc.ugr.es

Summary. Real optimization problems often involve not one, but multiple objectives, usually in conflict. In single-objective optimization there exists a global optimum, while in the multi-objective case no optimal solution is clearly defined, but rather a set of solutions, called Pareto-optimal front. Thus, the goal of multi-objective strategies is to generate a set of non-dominated solutions as an approximation to this front. This paper presents a novel adaptation of some of these meta-heuristics to solve the multi-objective Graph Partitioning problem.

Key words: Multi-Objective Optimization, Graph Partitioning, Simulated Annealing, Tabu Search

1 Introduction

Most real-life optimization problems require taking into account, not one, but multiple objectives simultaneously. In most cases, these objectives are in conflict, i.e., the improvement of some objectives imply the deterioration of others. Recently, the Pareto optimality concept [8] has been used to solve multi-objective optimization (MOO) problems.

Graph Partitioning is an important optimization problem that appears in many real life applications. It consist of obtaining a partition of the vertices of a graph into a given number of roughly equal parts, whilst ensuring that the number of edges connecting vertices in different sub-graphs is minimized. As the problem is NP-complete [4], efficient procedures providing high quality solutions in reasonable runtimes are very useful. Most of the proposed algorithms to solve the GPP are single-objective, i.e., the objective is to minimize the number of edges connecting vertices of different sub-graphs, while the imbalance is considered a constraint. Thus far, there exist few works related to GPP from the multi-objective perspective. This paper describes how to solve the graph partitioning problem by using Multi-Objective Meta-Heuristics (MOMHs).

Section 2 provides a more precise definition of the GPP, and offers a brief survey of the current state of the art. Section 3 describes the Pareto-based MOMHs we have adapted to solve the problem. Section 4 presents the results obtained by the different MOMHs in several test graphs, and the metrics used in the comparison. Finally, Section 5 contains the conclusions here obtained.

2 The Graph Partitioning Problem

The Graph Partitioning Problem (GPP) occurs in many real applications, as load balancing [18], VLSI design [6], task scheduling [1], etc.

Let G=(V, E) be a undirected graph, where V is the set of vertices, $|V|=n$, and E the set of edges which determines the connectivity of V. The GPP consists of dividing V into SG balanced sub-graphs, $V_1, V_2, \ldots, V_{SG}$, such that the number of edges that connect vertices belonging to different sub-graphs (cutsize) is minimized, verifying that $V_i \cap V_j = \phi$, $\forall$ $i \neq j$; and $\sum_{sg=1}^{SG} |V_{sg}| = |V|$. The balance constraint is defined by the maximum sub-graph weight, M=max($|V_{sg}|$), $\forall sg \in [1, SG]$.

In single-objective GPP, the objective is to minimize the cutsize, while the imbalance defined by M is considered a constraint. There exist many algorithms to solve the GPP in the single-objective perspective [9]. Most of them use the multilevel paradigm [11] in combination with any optimization technique. The main handicap of this model is the dependency on the imbalance constraint. Figure 1(a) shows a partition s, and Figure 1(b) other $s*$, which has been obtained by moving a boundary vertex from a sub-graph to the other. In Figure 1(c) both solutions are represented in the solution space. Thus, if the maximum allowed imbalance is x=30%, the solution s is selected as the best. However, if the maximum imbalance is x=20%, the best would be $s*$, due to s has a forbidden imbalance, despite having less cutsize.

Thus far, the number of works deal with the GPP is very limited [14, 15]. A typical approach is based on creating a mathematical multi-objective function as the weighting sum of the solution objective functions [15]. The main weakness of these approaches is the difficulty of assigning priorities to the objectives. An interesting way to overcome this disadvantage is to use the Pareto-dominance concept. Nevertheless, few works have focused on solving the multi-objective GPP by using the Pareto-dominance approach. We can reference [14], where authors adapted the SPEA multi-objective evolutionary algorithm [19] to the GPP. They tested the performance of this adaptation, obtaining unsatisfactory results. Their experimental results indicated that the use of a local search procedure allows for improvement in the quality of the solutions in comparison with the SPEA adaptation. This is the reason why we use MOMHs, discarding the MOEAs.

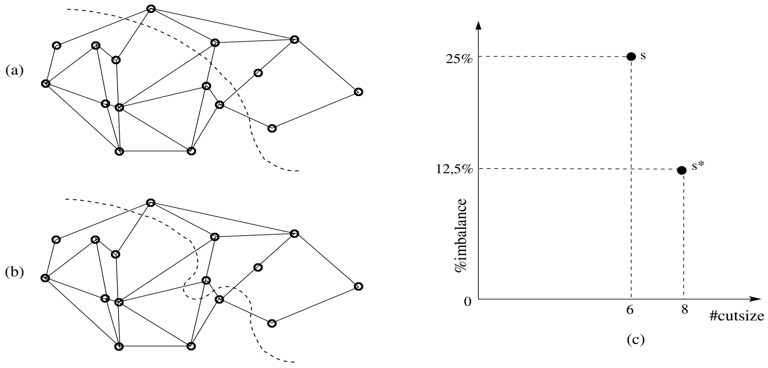


Fig. 1. Movement of s (a) to $s*$ (b), and their representation in the solution space.

3 Adapting Local Search MOMHs for GPP

Let P be a MOO problem, with $K \geq 2$ objectives. Instead of giving a scalar value to each solution, a partial order is defined based on Pareto-dominance relations. A solution s_1 is said to dominate another s_2 ($s_1 \prec s_2$) when it is better in at least one objective, and not worse in the others. A solution is said to be non-dominated if no solution can be found that dominates it. The set of non-dominated solutions is called Pareto optimal set. Since there is no single scalar decision, this set normally contains not one solution, but several. As all the objectives are equally important, the aim of multi-objective optimization is to induce this entire set.

Previous experiments [5, 2] have demonstrated the good behaviour of Simulated Annealing and Tabu Search in the single-objective GPP. These metaheuristics have been sucesfully applied in the multi-objective context. This section details the adaptation of some of these MOMHs to the multi-objective formulation of the GPP.

3.1 Serafini's Multi-Objective Simulated Annealing (SMOSA)

One of the first proposed MOMHs was Serafini's Multi-Objective Simulated Annealing (SMOSA) [16]. In single-objective Simulated Annealing (SA) [12] better neighboring solutions are always accepted, whereas worsening solutions are accepted by using the Metropolis function [13]. This mathematical function accepts new solutions with a certain probability, which is dependent on a parameter, called temperature. In a multi-objective context, it can be seen as always accepting a new solution ($s*$) if it dominates the current one (s), i.e. (s*≺s); and to accept $s*$ with a given probability if it is dominated by s, i.e. (s≺s*). However, there is a special case to consider when both solutions are indifferent, i.e., neither s dominates $s*$, nor $s*$ dominates s (s∼s*). Serafini suggested different transition probabilities for this case: some of these rules are used for the strong acceptance case, in which only dominating neighbors should be accepted with probability one, while other rule is used for the weak acceptance case, in which only dominated neighboring solutions should be accepted with a probability of less than one. Serafini demonstrated that the best choice is to combine strong and weak rules. The adaptation of SMOSA for the GPP is provided in Algorithm 1.

Algorithm 1 : Serafini's Multi-Objective Simulated Annealing (SMOSA)

```
Input: $G,RULE,\ T_i, T_{factor}, T_{stop}$;
s←GGA(G); cutsize[s]←eval_cutsize(s); imb[s]←eval_imbalance(s);
ND←$\phi$; t←T$_i$;
While (t>T$_{stop}$) do
    s*←movement(s);
    probability←Apply RULE(s*);
    If (Metropolis (probability,t)=1) then
        s←s*; Update ND;
    t←t*$Tfactor$;
Return ND;
```

3.2 Ulungu's Multi-Objective Simulated Annealing (UMOSA)

Another important SA-based MOMH is Ulungu's Multi-Objective Simulated Annealing (UMOSA) [17]. UMOSA executes separate runs by using fixed weights for each objective (see parameter λ in UMOSA algorithm). All the non-dominated sets obtained in each separate run are joined in a global set. Algorithm 2 provides the pseudo-code of UMOSA for GPP.

Algorithm 2 : Ulungu's Multi-Objective Simulated Annealing (UMOSA)

```
Input: $G,\ T_i,\ T_{factor},\ T_{stop},\ N_{runs}$;
For ($\forall$r $\in$ [1,N$_{runs}$]) do
    s←GGA(G); cutsize[s]←eval_cutsize(s); imb[s]←eval_imbalance(s);
    $\lambda_{cutsize}$←$\tau_{cutsize}$, $\lambda_{imb}$←$\tau_{imb}$; $\lambda_{cutsize}$, $\lambda_{imb}$$\in$[0,1], $\lambda_{cutsize}$+$\lambda_{imb}$=1.0;
    ND$_r$←$\phi$; t←T$_i$;
    While (t>T$_{stop}$) do
        s*←movement(s);
        $\Delta$s← ($\lambda_{cutsize}$ (cutsize[s*]-cutsize[s]) + $\lambda_{imb}$ (imb[s*]-imb[s]));
        If (Metropolis ($\Delta$s,t)=1) then
            s←s*; Update ND$_r$(s);
        t←t*$Tfactor$;
Global$_{ND}$← ND($\cup_{r=1}^{N_{runs}}$ ND$_r$);
Return Global$_{ND}$;
```

3.3 Czyzak's Pareto Simulated Annealing (PSA)

The third SA-based MOMH we have adapted is the Czyzak's Pareto Simulated Annealing (PSA) [3]. PSA is based on accepting neighboring solutions with a certain probability, which depends on the temperature parameter, as in UMOSA and SMOSA. However, while SMOSA and UMOSA use only a solution, PSA uses a set of solutions in the optimization process. In PSA, each solution dynamically modifies its weights in the objective function, which tries to assure adequate dispersion of the non-dominated solutions. It is important to indicate that a high weight associated with a given objective implies a lower probability of accepting new solutions that decrease the quality in this objective. Pseudocode of PSA is provided in Algorithm 3.

Algorithm 3 : Czyzak's Pareto Simulated Annealing (PSA)

```
Input: G, Population Size (ps), $T_i, T_{factor}, T_{stop}$, $\alpha$;
For ($\forall$s $\in$S) do
    s$\leftarrow$GGA(G); cutsize[s]$\leftarrow$eval_cutsize(s); imb[s]$\leftarrow$eval_imbalance(s);
ND$\leftarrow\phi$; $\leftarrow$T$_i$;
For ($\forall$s $\in$S) do
    Update ND(s);
While (t>T$_{stop}$)
  For ($\forall$s $\in$S) do
      s*$\leftarrow$ movement(s);
      If (s$\not\prec$s*) then
          Update ND(s*);
      s'$\leftarrow$closest solution to s, such that f(s)$\not\prec$f(s'), s'$\in$S;
      If ($\nexists$s') then
          $\lambda_{cutsize}\leftarrow \tau_{cutsize}$, $\lambda_{imb}\leftarrow \tau_{imb}$; $\lambda_{cutsize}$, $\lambda_{imb}\in$[0,1], $\lambda_{cutsize}+\lambda_{imb}$=1.0;
      Else then
          If (cutsize[s]$\prec$cutsize[s']) then $\lambda_{cutsize}\leftarrow\alpha\lambda_{cutsize}$;
          If (cutsize[s']$\prec$cutsize[s]) then $\lambda_{cutsize}\leftarrow\alpha/\lambda_{cutsize}$;
          If (imb[s]$\prec$imb[s']) then $\lambda_{imb}\leftarrow\alpha\lambda_{imb}$;
          If (imb[s']$\prec$imb[s]) then $\lambda_{imb}\leftarrow\alpha/\lambda_{imb}$;
          Normalize $\lambda$ such that $\sum_{k=1}^{K}\lambda_k$=1.0;
      $\Delta$s$\leftarrow$ ($\lambda_{cutsize}$ (cutsize[s*]-cutsize[s]) + $\lambda_{imb}$ (imb[s*]-imb[s]));
      If (Metropolis ($\Delta$s,t)=1) then
          s$\leftarrow$s*; Update ND$_i$; N$_{count}\leftarrow$0;
  t$\leftarrow$t*$Tfactor$;
Return ND;
```

3.4 Hansen's Multi-Objective Tabu Search (MOTS)

Tabu Search (TS) [7] is other local search-based optimization method which repeatedly moves from a current solution to the best in the neighborhood, while trying to avoid being trapped in local optimum by maintaining a list of tabu moves. Hansen proposed Multi-Objective Tabu Search (MOTS) [10], which works with a set of solutions which are improved by using a TS acceptance criterion. In MOTS, the weighting values are adaptively modified during the search process. The number of solutions changes according to the dominance rank of each solution. Thus, if the rank is high, the solutions overlap each other in the objective space, and then the number of solutions is decreased. However, a low average rank indicates that the non-dominated solutions are well-spread. The adaptation of MOTS is showed in Algorithm 4.

```
Algorithm 4 : Hansen's Multi-Objective Tabu Search (MOTS)
  Input: G, Initial Population Size (ps), $N_{stop}$;
  For ($\forall$s $\in$S) do
       s$\leftarrow$GGA(G); cutsize[s]$\leftarrow$eval_cutsize(s); imb[s]$\leftarrow$eval_imbalance(s); $TL_s \leftarrow \phi$;
 ND$\leftarrow \phi$; $N_{count} \leftarrow 0$; $\pi_{cutsize} \leftarrow \pi_{imb} \leftarrow 0.5$;
  While ($N_{count} < N_{stop}$) do
      For (each solution s$\in$S) do
          $\lambda_{cutsize} \leftarrow \lambda_{imb} \leftarrow 0$;
          For (each solution s*$\in$S, s$\not\prec$s*, $\forall$s*$\neq$s) do
               w$\leftarrow$1/distance(s,s*);
               If(cutsize[s]>cutsize[s*]) do
                  $\lambda_{cutsize} \leftarrow \lambda_{cutsize} + \pi_{cutsize}$w;
               If(imb[s]>imb[s*]) do
                  $\lambda_{imb} \leftarrow \lambda_{imb} + \pi_{imb}$w;
      If ($\lambda_{cutsize} = \lambda_{imb} = 0$) then
          $\lambda_{cutsize} \leftarrow \tau_{cutsize}$, $\lambda_{imb} \leftarrow \tau_{imb}$; $\lambda_{cutsize}$, $\lambda_{imb} \in [0,1]$, $\lambda_{cutsize} + \lambda_{imb} = 1.0$;
      Find s* which maximizes $\lambda$f(s*), s*$\in$Neighborhood(s), movement(s,s*)$\notin TL_s$;
      If ($TL_s$ is full) then remove oldest element of $TL_s$;
      s$\leftarrow$s*; $TL_s \leftarrow TL_s \cup$movement(s,s*);
      If ($\nexists$s'$\in$ND such as s'$\prec$s) then ND$\leftarrow$ND$\cup$s, and update $\pi$;
      $N_{count} \leftarrow N_{count}$+1 ;
 Return ND;
```

4 Experimental Results

The executions have been performed by using a 2.4 GHz processor. The graphs used to evaluate the MOMHs (Table 1) belong to a public domain set [9] frequently used to compare single-objective graph partitioning algorithms.

This table details the number of vertices and edges, maximum connectivity (*max*), minimum connectivity (*min*), and average connectivity (*avg*).

Table 1. Test Graphs

graph	|V|	|E|	min	max	avg
add20	2395	7462	1	123	6.23
3elt	4720	13722	3	9	5.81
uk	4824	6837	1	3	2.83
add32	4960	9462	1	31	3.82
crack	10240	30380	3	9	5.93
wing_nodal	10937	75488	5	28	13.80

4.1 Parameter Setting

As we commented above, SMOSA and UMOSA use a single solution to perform the search process, while PSA y MOTS use a set of solutions. In both cases, the population size has been set to |P|=35. The initial solutions are obtained by using the Graph Growing Algorithm (GGA) [11]. The maximum size of the non-dominated set has also been set to |ND|=35. The annealing parameters for SMOSA, UMOSA, and PSA are Ti=100, and Tfactor=0.999. In UMOSA, the number of runs with different weighting values has been set to 10 ($\lambda_{cutsize}$=1.0 ... $\lambda_{cutsize}$=0.0). Note that λ_{imb}=1.0-$\lambda_{cutsize}$. The number of iterations has been set to 11500. This value corresponds to the number of iterations needed by the SA-based MOMHs to fall below t=0.01 with these annealing values. The results here shown correspond to the partitioning of the test graphs into SG=16 sub-graphs. In this work, we have considered an additional constraint in order to discard those solutions with an imbalance greater than 5% (i.e., $M\leq$1.05).

4.2 Metrics used to evaluate the quality of the solutions

As the GPP is NP-complete, the exactly location of the Pareto-optimal set cannot be determined. In order to compare the quality of the non-dominated solutions, we propose four metrics we describe in what follows:

- Metric *M1*: size of the non-dominated set.
$$M1 \leftarrow |\mathrm{ND}| \quad (1)$$

- Metric *M2*: average area under the non-dominated solutions, taking as reference the maximum cutsize of the initial solutions.
$$M2 \leftarrow \sum_{s=1}^{|ND|}\left(\frac{\#cutsize[s]}{\#max\_cutsize} * \frac{\#imbalance[s]}{\#max\_imbalance}\right)/|ND|;\ \forall \mathrm{s} \in ND \quad (2)$$

- Metric *M3*: extension of the non-dominated set.

$$M3 \leftarrow \max\{\text{distance(s,s')}\}; \quad \forall \text{ s,s'} \in \text{ND, s}\neq\text{s'} \quad (3)$$

- Metric *M4*: penalty associate to the crowding degree.

$$M4 \leftarrow \sqrt[|ND|]{\prod_{s=1}^{|ND|} distance(s,s')} \text{ ; } \forall\text{s,s'}\in\text{ND,}s\text{'=nearest to } s \quad (4)$$

4.3 Analysis of the results

Table 2 shows the results obtained by the MOMHs in the partitioning of the graphs described in Table 1 into SG=16 subgraphs. In reference to the metric *M1*, PSA often obtain the non-dominated sets with a greater number of solutions than the other MOMHs. The smallest average area under the non-dominated set (metric *M2*) is obtained by UMOSA. This indicates that the quality of the solutions obtained by UMOSA is superior in most cases to the other MOMHs. SMOSA obtains (in average) non-dominated sets more extensive than the other MOMHs (metric *M3*). This table also shows that the dispersion of non-dominated sets (*M4*) is clearly better in UMOSA. Finally, last column shows that SA-based MOMHs obtain similar runtimes, although in PSA it is higher because it optimizes a set of solutions (as MOTS). Thus, we can conclude that, although the population based MOMHs obtain larger non-dominated sets, the quality of them is higher in SMOSA and UMOSA.

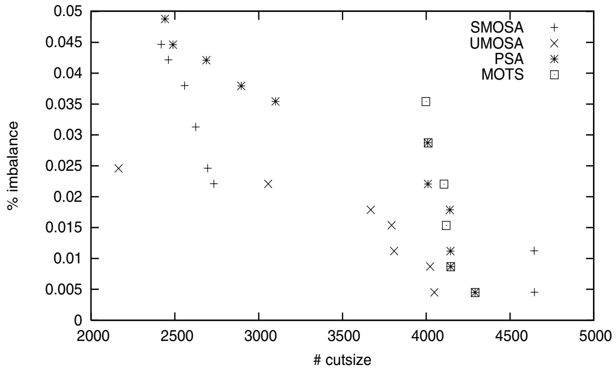


Fig. 2. Evaluating the MOMHs in add20

In addition to the numerical analysis, Figure 2 shows the non-dominated solutions obtained by the MOMHs in the partitioning of *add20* into *SG*=16 sub-graphs. PSA obtains the greater non-dominated set, while UMOSA obtains the non-dominated set with the lowest average enclosed area. On the other hand, PSA and SMOSA obtain the more extensive non-dominated sets. Finally, the smaller average crowding degree is obtained by UMOSA.

Table 2. Comparing MOMHs: SMOSA,MOSA,PSA,MOTS

graph	Algorithm	M1	M2	M3	M4	runtime (secs.)
add20	SMOSA$_M$	8	0.319	0.660	0.050	125
	UMOSA	7	0.228	0.420	0.101	606
	PSA	11	0.359	0.683	0.056	400
	MOTS	6	0.325	0.439	0.082	11194
3elt	SMOSA$_M$	6	0.047	0.682	0.060	185
	UMOSA	2	0.005	0.614	0.614	316
	PSA	4	0.065	0.252	0.057	457
	MOTS	6	0.195	0.734	0.065	5250
uk	SMOSA$_M$	4	0.043	0.599	0.072	143
	UMOSA	3	0.039	0.476	0.054	258
	PSA	10	0.144	0.492	0.050	309
	MOTS	4	0.101	0.253	0.077	5352
add32	SMOSA$_M$	4	0.003	0.703	0.079	137
	UMOSA	5	0.045	0.706	0.093	315
	PSA	15	0.377	0.892	0.050	453
	MOTS	12	0.291	0.522	0.047	7822
crack	SMOSA$_M$	3	0.036	0.257	0.083	539
	UMOSA	2	0.004	0.026	0.026	565
	PSA	13	0.100	0.314	0.025	915
	MOTS	4	0.023	0.069	0.023	23522
wing_nodal	SMOSA$_M$	9	0.058	0.416	0.013	1003
	UMOSA	3	0.010	0.521	0.020	662
	PSA	6	0.185	0.260	0.020	2083
	MOTS	6	0.299	0.494	0.013	41841
average	SMOSA$_M$	5.66	0.084	**0.553**	0.06	**355**
	UMOSA	3.67	**0.055**	0.461	**0.151**	454
	PSA	**9.83**	0.205	0.482	0.043	770
	MOTS	6.33	0.206	0.419	0.050	15830

5 Conclusions

In this paper, we have presented a novel adaptation of MOMHs for the multi-objective GPP. This is one of the few papers that compare these techniques, and the first one which applies them to this problem. Previous adaptations of MOEAs, like SPEA, to this problem have obtained dissapointing results. The results obtained by these MOMHs over several test graphs indicate that the population-based MOMHs (MOTS and PSA) obtain the most extensive sets of non-dominated solutions. On the other hand, UMOSA and SMOSA often obtain high quality solutions in comparison with the other techniques. Aspects to analyze are the hybridization of some of these MOMHs, and the parallelization of the population-based versions.

Acknowledges

Work supported by the Spanish MCyT under contracts TIC2002-00228. Authors appreciate the support of the "Structuring the European Research Area" programme, R113-CT-2003-506079, funded by the European Commission.

References

1. Aleta A, Codina JM, Sanchez J, Gonzalez A (2001) Graph-Partitioning Based Instruction Scheduling for Clustered Processors. Proc. 34th ACM/IEEE International Symposium on Microarchitecture, pp 150-159
2. Baños R, Gil C, Ortega J, Montoya FG (2004) Parallel Multilevel Metaheuristic for Graph Partitioning, J. of Heuristics, 10(3):315-336
3. Czyzak P, Jaszkiewicz A (1998) Pareto Simulated Annealing - a Metaheuristic Technique for Multiple-objective Combinatorial Optimization, J. of Multi-Criteria Decision Analysis 7:34-47
4. Garey MR, Johnson DS (1979) Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman & Company, San Francisco
5. Gil C, Ortega J, Montoya MG, Baños R (2002) A Mixed Heuristic for Circuit Partitioning, Computational Optimization and Applications J. 23(3):321-340
6. Gil C, Ortega J, Montoya MG (2000) Parallel VLSI Test in a Shared Memory Multiprocessors, Concurrency: Practice and Experience, 12(5):311-326
7. Glover F, Laguna M (1993) Tabu Search. In: C.R. Reeves (eds) Modern Heuristic Techniques for Combinatorial Problems. Blackwell, London, pp. 70-150
8. Goldberg DE (1989) Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley
9. Graph Partitioning Arch. http://staffweb.cms.gre.ac.uk/∼c.walshaw/partition/
10. Hansen PH (1997) Tabu Search for Multiobjective Optimization: MOTS. Proc. of the 13th International Conference on Multiple Criteria Decision Making
11. Karypis G, Kumar V (1998) A Fast High Quality Multilevel Scheme for Partitioning Irregular Graphs. Tech. Report TR-95-035, University of Minnesota
12. Kirkpatrick S, Gelatt CD, Vecchi MP, (1983) Optimization by Simulated Annealing. Science 220(4598):671-680
13. Metropolis N, Rosenbluth A, Rosenbluth M, Teller A, Teller E, (1953) Equation of State Calculations by Fast Computing Machines. J. of Chemical Physics, 21(6):1087-1092
14. Rummler A, Apetrei A (2002) Graph Partitioning Revised - a Multiobjective Perspective. Proc. of 6th MultiConf. On Syst., Cybern. and Informatics
15. Selvakkumaran N, Karypis G (2003) Multi-Objective Hypergraph Partitioning Algorithms for Cut and Maximum Subdomain Degree Minimization. Proc. Int. Conference on Computer Aided Design, pp 726-733
16. Serafini P. (1993) Simulated Annealing for Multi-Objective Optimization problems. In G.H. Tzeng et al. (eds) Multiple Criteria Decision Making. Expand and Enrich the Domains of Thinking and Application, Springer Berlin, pp. 283-292
17. Ulungu E, Teghem J, Fortemps P, Tuyytens D (1999) MOSA Method: A Tool for Solving Multiobjective Combinatorial Optimization Problems. J. of Multi-Criteria Decision Analysis 8(4):221-236
18. Walshaw C, Cross M, Everett MG (1999) Mesh Partitioning and Load-balancing for Distributed Memory Parallel Systems. In: B. Topping (eds) Parallel & Distributed Proc. for Computational Mechanics: Systems and Tools, pp. 110-123
19. Zitzler E, Thiele L (1999) Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Transactions on Evolutionary Computation 3(4):257-271

Part IV

Diagnosis and Fault Tolerance

Genetic Algorithms for Artificial Neural Net-based Condition Monitoring System Design for Rotating Mechanical Systems

Abhinav Saxena

School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta,
GA 30332, USA.

Ashraf Saad, PhD

Associate Professor, School of Electrical and Computer Engineering, Georgia Institute of Technology, Savannah,
GA 31407, USA.

Abstract. We present the results of our investigation into the use of Genetic Algorithms (GA) for identifying near optimal design parameters of Diagnostic Systems that are based on Artificial Neural Networks (ANNs) for condition monitoring of mechanical systems. ANNs have been widely used for health diagnosis of mechanical bearing using features extracted from vibration and acoustic emission signals. However, different sensors and the corresponding features exhibit varied response to different faults. Moreover, a number of different features can be used as inputs to a classifier ANN. Identification of the most useful features is important for an efficient classification as opposed to using all features from all channels, leading to very high computational cost and is, consequently, not desirable. Furthermore, determining the ANN structure is a fundamental design issue and can be critical for the classification performance. We show that GA can be used to select a smaller subset of features that together form a genetically fit family for successful fault identification and classification tasks. At the same time, an appropriate structure of the ANN, in terms of the number of nodes in the hidden layer, can be determined, resulting in improved performance.

Keywords: Genetic Algorithms, Artificial Neural Networks, Hybrid Techniques, Fault Diagnosis, Condition Monitoring, Rotating Mechanical Systems.

Introduction

With the increase in production capabilities of modern manufacturing systems, plants are expected to run continuously for extended hours. Consequently, unexpected downtime due to machinery failure has become more costly than ever before and therefore condition monitoring is gaining importance in industry because of the need to increase machine availability and health trending, to warn of impending failure, and/or to shut down a machine to prevent further damage. It is required to detect, identify and then classify different kinds of failure modes that can occur within a machine system. Often several different kinds of sensors are employed at different positions to sense a variety of possible failure modes. Features are then calculated to analyze the signals from all these sensors to assess the health of the system.

Significant research on fault detection in gears and bearings has been carried out so far and has resulted in the identification of a rich feature library with a variety of features from the time, frequency and wavelet domains. However, the requirements in terms of suitable features may differ depending on how these elements are employed inside a complex plant. Moreover, there are several other systems like planetary gears that are far more complex and a good set of features for them has not yet been clearly identified. Due to some structural similarities, it is natural to search for suitable features from existing feature libraries. An exhaustive set of all features that captures a variety of faults becomes very large for practical purposes. A feature selection process must be identified in order to speed up computation and to also increase the accuracy of classification. Genetic Algorithms (GA) offer suitable means to do so, given the vast search space formed by the number of possible combinations of all available features in a typical real-world application. GA-based techniques have also been shown to be extremely successful in evolutionary design of various classifiers, such as those based on Artificial Neural Networks (ANNs). Using a reduced number of features that primarily characterize the system conditions along with optimized structural parameters of ANNs have been shown to give improved classification performance (Jack and Nandi 2001, Samanta, 2004a, 2004b). The work presented herein uses GA for obtaining a reduced number of good features in addition to determining the parameters of the classifier ANN. Our results show the effectiveness of the extracted features from the acquired raw and preprocessed signals in diagnosis of machine condition.

Using Genetic Algorithms and ANNs

Feature selection is an optimization problem of choosing an optimal combination of features from a large and possibly multimodal search space. Two major classes of optimization techniques have traditionally been used, namely: calculus-based techniques, that use gradient-based search mechanisms, and enumerative techniques, such as dynamic programming (DP) (Tang *et al.* 1996). For an ill-defined or multimodal objective function where a global minimum may not be possible or very difficult to achieve, DP can be more useful but its computational complexity makes it unsuitable for effective use in most practical cases. Thus a non-conventional nonlinear search algorithm is desired to obtain fast results, and GA meets these requirements. From another standpoint the basic problem here is that of high dimensionality, with a large number of features among which it is not known which ones are good for identifying a particular fault type. There are several methods that can be used to reduce the dimensionality of the problem. Principle Component Analysis (PCA) is the most widely used technique among similar techniques such as Multi Dimensional Scaling (MDS) and Singular Value Decomposition (SVD) (Vlachos *et al.* 2002). Other methods are low dimensional projection of the data (projection pursuit, generalized additive models) (Miguel 1997), regression (principle curves), and self-organization (Kohonen maps) just to name a few. PCA can be much less computationally expensive than a GA-based approach. However, all features need to be computed for PCA before a rotated feature space can be created and therefore it still requires computation of all features demanding a large amount of data processing. GA facilitates a better scenario, in which although the computational cost will be very high during the offline training and feature selection phase, much less computing is required for online classification. Other methods for feature selection include Forward Selection that assesses almost all combinations of different features that are available to determine a set of best features (Dallal 2004). The main difficulty in applying Forward Selection occurs when two features individually perform poorly, but lead to better results when used together. The pair of such good features found by Forward Selection may not necessarily be the best combination, as it chooses the candidates based on their respective individual performances. GA-based methods can be used to add more functionality in parameter selection. For instance, it can be used to simultaneously find the optimal structure of an ANN, in terms of concurrently determining the number of nodes in the hidden layers and the connection matrices for evolving the ANNs

(Balakrishnan and Honavar 1995, Edwards *et al.* 2002, Filho *et al.* 1997, Sunghwan and Dagli 2003). Potential applications of ANNs in automated detection and diagnosis have been shown in (Jack and Nandi 2001, Samanta 2004a, Shiroishi e*t al.* 1996). Similar optimization can be considered irrespective of what classification technique is used, an evolving search is expected to provide a better combination, especially in the cases where the dimensionality increases the possible number of combinations exponentially, and hence the computational power needed. Approaches using Support Vector Machines and Neuro-Fuzzy Networks have also been put forward for solving the feature selection problem (Samanta 2004a), but the use of GA still remains warranted.

Genetic Algorithms

In 1975, Holland introduced an optimization procedure that mimics the process observed in natural evolution called Genetic Algorithms (Holland 1975). GA is a search process that is based on the laws of natural selection and genetics. As originally proposed, a simple GA usually consists of three processes *Selection, Genetic Operation* and *Replacement.* Figure 1 shows a typical GA cycle.

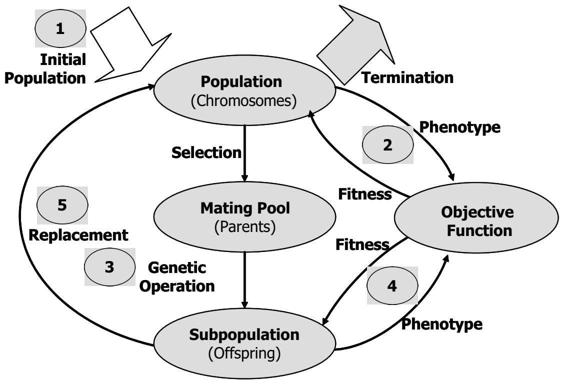


Fig. 1. Genetic algorithm life-cycle

The population comprises a group of chromosomes that are the candidates for the solution. The fitness values of all chromosomes are evaluated using an objective function (performance criteria or a system's behavior) in a decoded form (phenotype). A particular group of *parents* is selected from the population to generate *offspring* by the defined genetic operations of *crossover* and *mutation*.

The fitness of all offspring is then evaluated using the same criterion and the chromosomes in the current population are then replaced by their offspring, based on a certain replacement strategy. Such a GA cycle is repeated until a desired termination criterion is reached. If all goes well throughout this process of simulated evolution, the best chromosome in the final population can become a highly evolved solution to the problem.

Artificial Neural Networks

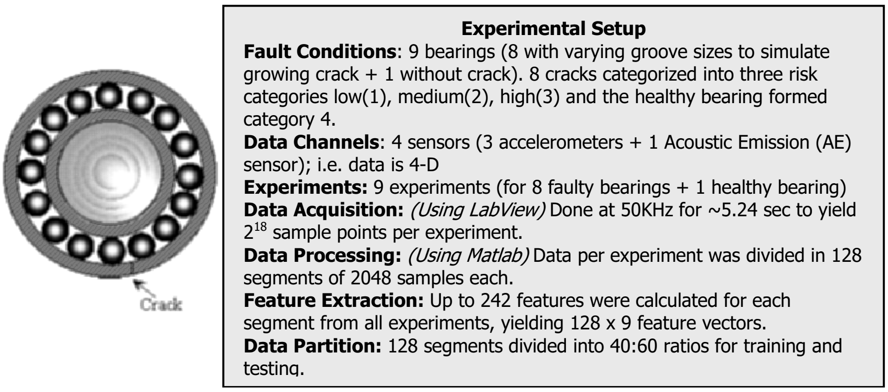
An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. A great deal of literature is available explaining the basic construction and similarities to the biological neurons. The discussion here is limited to a basic introduction of several components involved in the ANN implementation. The network architecture or topology (including: number of nodes in hidden layers, network connections, initial weight assignments, activation functions) plays a very important role in the performance of the ANN, and usually depends on the problem at hand. In most cases, setting the correct topology is a heuristic model selection. Whereas the number of input and output layer nodes is generally suggested by the dimensions of the input and the output spaces. Too many parameters lead to poor generalization (over fitting), and too few parameters result in inadequate learning (under fitting) (Duda *et al.* 2001). Some aspects of ANNs are described next.

Every ANN consists of at least one *hidden layer* in addition to the input and the output layers. The number of hidden units governs the expressive power of the net and thus the complexity of the decision boundary. For well-separated classes fewer units are required and for highly interspersed data more units are required. The number of synaptic weights is based on the number of hidden units representing the degrees of freedom of the network. Hence, we should have fewer weights than the number of training points. As a rule of thumb, the number of hidden units is chosen as *n/10*, where *n* is the number of training points (Duda *et al.* 2001, Lawrence *et al.* 1997). But this may not always hold true and a better tuning might be required depending on the problem.

Network Learning: Input patterns are exposed to the network and the network output is compared to the target values to calculate the error, which is corrected in the next pass by adjusting the synaptic weights. Several training algorithms have been designed; the most commonly used being the Levenberg-Marquardt (LM) back-propagation algorithm which is an extension of LMS algorithms for linear systems. However, resilient back propagation has been shown to work faster on larger networks (Riedmiller 1993), and has thus been used throughout this study. Training can be done in different ways. In *Stochastic Training* inputs are selected stochastically (randomly) from the training set. *Batch Training* involves presenting the complete training data before weights are adjusted, and thus several epochs are required for effective training. For *Incremental Training* weights are adjusted after each pattern is presented. ***Stopping Criterion*** indicates when to stop the training process. It can be a predetermined limit of absolute error, Minimum Square Error (MSE) or just the maximum number of training epochs.

Problem Description and Methods

A simpler problem of a roller bearing health monitoring has been used to illustrate the effectiveness of GA in feature selection for fault classification using ANNs. Several bearings with different outer race defects (Figure 2) were used in the test setup.



Experimental Setup

Fault Conditions: 9 bearings (8 with varying groove sizes to simulate growing crack + 1 without crack). 8 cracks categorized into three risk categories low(1), medium(2), high(3) and the healthy bearing formed category 4.

Data Channels: 4 sensors (3 accelerometers + 1 Acoustic Emission (AE) sensor); i.e. data is 4-D

Experiments: 9 experiments (for 8 faulty bearings + 1 healthy bearing)

Data Acquisition: *(Using LabView)* Done at 50KHz for ~5.24 sec to yield 2^{18} sample points per experiment.

Data Processing: *(Using Matlab)* Data per experiment was divided in 128 segments of 2048 samples each.

Feature Extraction: Up to 242 features were calculated for each segment from all experiments, yielding 128 x 9 feature vectors.

Data Partition: 128 segments divided into 40:60 ratios for training and testinq.

Fig. 2. (a) Outer race roller bearing defect. (b) Experimental Setup

Defects were constructed using a die sinking Electrical Discharge Machine (EDM) to simulate 8 different crack sizes resulting in 9 health conditions; 8 faulty and one healthy. The groove geometries used in these tests have been described in (Billington 1997). In all, four sensors (three accelerometers and one acoustic emission sensor) were employed to get the signal from all eight faulty bearings plus one bearing without any defect. These sensors were attached to signal conditioners and a programmable low pass filter such that each had ground-isolated outputs. The defect sizes were further categorized in four groups including a no-defect category. All tests were run at a radial load of 14730 N and a rotational speed of 1400 RPM. The data was acquired in the time domain as several snapshots of the time series for each case.

Feature Extraction

As mentioned earlier, published research has been conducted on diagnostic feature extraction for gears and bearing defects. Statistical features provide a compact representation of long time series data. A great deal of information regarding the distribution of data points can be obtained using first, second and higher order transformations of raw sensory data, such as various moments and cumulants of the data. Features such as mean, variance, standard deviation, and kurtosis (normalized fourth central moment) are the most common features employed for rotating mechanical components. It is also worthwhile sometimes to explore other generalized moments and cumulants for which such common names do not exist. An approach suggested in (Jack and Nandi 2001) has been adapted to create a variety of such features in a systematic manner. Four sets of features were obtained based on the data preprocessing and feature type. These features were calculated from the data obtained from all sensors for all fault levels.

1) *Plain statistical features:* A number of features were calculated based on moments ($m_x^{(i)}$) and cumulants ($C_x^{(i)}$) of the vibration data obtained from the sensors. Where the i^{th} moment $m_X^{(i)}$ of a random variable X is given by the expectation E(X^i) and the cumulants $C_x^{(i)}$ are defined as described below in equations 1-4:

$$C_x^{(1)} = m_x^{(1)} \tag{1}$$

$$C_x^{(2)} = m_x^{(2)} - (m_x^{(1)})^2 \tag{2}$$

$$C_x^{(3)} = m_x^{(3)} - 3\, m_x^{(2)} m_x^{(1)} + 2(m_x^{(1)})^3 \tag{3}$$

$$C_x^{(4)} = m_x^{(4)} - 3(m_x^{(2)})^2 - 4\, m_x^{(3)} m_x^{(1)} + 12 m_x^{(2)} (m_x^{(1)})^2 - 6(m_x^{(1)})^4 \tag{4}$$

$$w = \bullet(x^2 + y^2 + z^2) \tag{5}$$

$$\begin{aligned}\underline{v} = [\quad & m_x^{(1)}\, m_y^{(1)}\, m_z^{(1)} \\ & C_x^{(2)}\, C_y^{(2)}\, C_z^{(2)}\, C_x^{(1)}{*}C_y^{(1)}\, C_y^{(1)}{*}C_z^{(1)}\, C_z^{(1)}{*}C_x^{(1)}\, C_x^{(3)}\, C_y^{(3)}\, C_z^{(3)}\, C_x^{(1)}{*}C_y^{(2)}\, C_x^{(2)}{*}C_y^{(1)} \ldots \\ & C_y^{(1)}{*}C_z^{(2)}\, C_y^{(2)}{*}C_z^{(1)}\, C_z^{(1)}{*}C_x^{(2)}\, C_z^{(2)}{*}C_x^{(1)} \\ & C_x^{(4)}\, C_y^{(4)}\, C_z^{(4)}\, C_x^{(1)}{*}C_y^{(3)}\, C_x^{(2)}{*}C_y^{(2)}\, C_x^{(3)}{*}C_y^{(1)} \ldots \\ & C_y^{(1)}{*}C_z^{(3)}\, C_y^{(2)}{*}C_z^{(2)}\, C_y^{(3)}{*}C_z^{(1)} \ldots\end{aligned}$$

$$\begin{array}{l} C_z^{(1)}*C_x^{(3)}\ C_z^{(2)}*C_x^{(2)}\ C_z^{(3)}*C_x^{(1)} \\ m_w^{(1)}\ C_w^{(2)}\ C_w^{(3)}\ C_w^{(4)} \\ m_a^{(1)}\ C_a^{(2)}\ C_a^{(3)}\ C_a^{(4)} \qquad\qquad]^T \end{array} \tag{6}$$

Three accelerometers were used to measure the individual vibration components in three directions x, y and z. In order to compute the combined effect another signal *w* was generated (Equation 5). Cumulants, as described in Equations 1-4 were computed for all *x, y, z, w,* and *a* (acoustic emission) signals, and a 38 element feature vector was formed as described by Equation 6. In total, 128 snapshot segments from each of the nine experiments were used to form a sequence of 1152 (128x9) sample points. Each sample point consists of four rows of time series data from the four sensors, and the subsequent feature calculation yields a 38-element feature vector as determined by Equation 6. Thus a 1152x38 feature matrix was obtained for conducting the experiments with statistical features.

2) *Signal Differences and Sums:* As a further preprocessing on raw data, sum and difference signals were calculated for all 4 sensor channels in order to highlight high and low frequency content of the raw signals, as shown in Equations 7 and 8. Difference signals should increase whenever the high frequency changes take place, and the sum signals would show similar effects for the low frequency content.

$$d(n) = x(n) - x(n\text{-}1) \tag{7}$$

$$i(n) = \{\ x(n) - m_x^{(1)}\ \} + i(n\text{-}1) \tag{8}$$

where $m_x^{(1)}$ is the mean of the sequence *x.*

Equations 1-6 were applied to both *d(n)* and *i(n)* to create two more 38x1152 feature matrices. Since these sum and difference signals consist of only first order derivatives, computing them is fairly fast for the online processing requirements, and hence are suitable candidates as signals for feature calculation. Moreover several sensors are capable of computing these measurements internally (using hardware filtering) thereby eliminating the requirement of further data processing.

3) *Spectral Features:* Frequency domain features are very informative for rotating components, since well- defined frequency components are associated with them. Any defect associated with the balls or the inner race of bearings expresses itself as a high frequency component. For each of the four channels, a 64-point Fast Fourier Transform (FFT) was carried out. Based on visual inspection, or simple threshold test, it was observed that the frequency components were relatively small in magnitudes beyond the first 32 values. Consequently, only the first 32 values from each channel

(i.e., a total of 128 values from all four channels) were retained. Although this could be done automatically using some preset threshold, a fixed value was used for this study. This results in another 128x1152 feature matrix for spectral features. Although other features can be computed based on FFT of the raw signal, these were considered sufficient in order to show the effectiveness of GA in selecting the best features among the ones available.

Five feature sets were defined: 1) statistical features on raw signal (38 values); 2) statistical features on sum signal (38 values); 3) statistical features on difference signals (38 values); 4) spectral features (128 values), and 5) all the features considered together (242 values). The raw data was normalized using Equation 9 prior to the feature calculation. It has been shown that the normalized data performs much better in terms of training time and training success (Jack and Nandi 2001)**.** This helps in fast and uniform learning of all categories and results in small training errors (Duda *et al.* 2001).

$$x_i = (x_i - \mu_x)/ {}_x \qquad (9)$$

Where μ_x is the mean and ${}_x$ is the variance of the sequence x

As can be realized, a large feature set was obtained with only limited feature calculation techniques and a sensor suit mounted only at one location. In practice several such sensor suites are mounted at different locations to monitor for the occurrence of various faults, and this further increases the dimensionality of the problem. Almost all ANN-based classification techniques would take a long time to train such large networks and may not still achieve a good performance. Forming all combinations of a reduced feature set and testing them exhaustively is practically impossible. Further, searching for an optimal structure of the ANN increases the complexity of the search space considerably. In such a vast search space GA is the most appropriate non-linear optimization technique. The details of implementation for this research are described below.

Implementation

The experiments were conducted using an actual test bed and the computer implementation was done using MATLAB ver.6.5 on a PC with Pentium Celeron 2.2 GHz processor and 512 MB RAM.

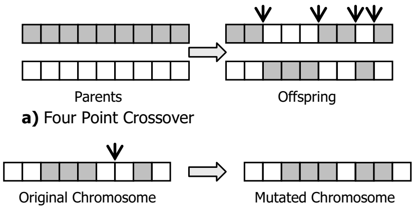
Chromosome encoding: A binary chromosomal representation was adopted. The length of the chromosome depends directly on the number of features required in the solution set for inputs to the ANN. For this study, this number was fixed at 10. This number is usually defined by the amount of time and computational power available. However, a GA-based search augmented with additional constraints and cost functions for fitness evaluation can be used to dynamically find an optimal value for the number of features to use, rather than pre-specifying it explicitly. Since the selection should be made from 38 (76, 128 or 242 depending on the feature set used) values, each gene consists of 6 (7, 7 or 8) bits (respectively). As a result, the total length of the chromosome becomes 60 (6x10) and likewise (70, 70 or 80, respectively) in other cases. In order to take care of genotypes without meaningful phenotypes, the numbers generated were wrapped around the maximum number of features in the corresponding feature set. Several experiments were carried out in order to find an appropriate population size using trial and error. It was found that a population size of 20 was appropriate for convergence in reasonable time and was therefore kept fixed at 20 for all of the experiments. Later it was realized that an integer coding of size ten could be a faster and more efficient method and also take care of genotypes without meaningful phenotypes.

Crossover: We implemented a multi-point crossover. The location of crossover points is determined randomly. All members of the population were made to crossover after they were paired based on their fitness. The chromosomes are then sorted in the order of decreasing fitness, and pairs were formed with immediate neighbors; i.e., the best chromosome was paired with the next best and the third best chromosome with the next one on the list, and so on. Figure 3a illustrates a 4-point crossover.

Mutation: A multipoint bit-flip mutation based on a pre-specified probability of mutation (p_m) was implemented. Again the location of mutation is randomly determined every time the mutation operator is applied to a chromosome. Figure 3b illustrates a single point mutation.

Fitness Function: Two fitness functions were used to capture the system performance. The number of correct classifications by the ANN was used as the first fitness function. The output of the ANN was expected to be 1, 2, 3 or 4 in value corresponding to one of four different fault levels, based on which fault level is recognized. The ANN output was rounded to the nearest integer to make a decision as shown in Figure 3c. This represents the classification success of the ANN and rejects the magnitude of training error. However, in order to evaluate the performance in terms of training efficiency, the second criterion used is the total absolute error. The total absolute error represents the accuracy of the ANN training. No direct penalty was made for incorrect classifications. However, other fitness functions can also be implemented that might involve mean squared error or some other penalty scheme over incorrect classifications.

After a new generation of offspring is obtained, the fitness of all chromosomes (both parents and offspring) is evaluated, and the ones with the highest fitness are carried to the next generation for the next genetic cycle. Two experiments were conducted in order to compare the performance of standalone ANNs and GA-supported ANNs.



a) Four Point Crossover

b) Single point bit-flip mutation

Total Absolute Error
Summation of the difference between the ANN output and Actual target value for each data point
$\Sigma \mid z(i) - t(i) \mid$; $z(i)$ = ANN output, $t(i)$ = target value

Number of Correct Classifications
If $round\{z(i)\} - t(i)\} = 0$; consider it a hit.
Count all the hits and return as fitness value

c) Fitness evaluation criteria

Fig. 3. Basic building blocks and design parameters in GA implementation

First, the comparison was made between the performance of a fixed size ANN with and without GA optimization. Thus an ANN with all features used as input is compared to an ANN where the 10 best features selected by GA are used as input. The number of hidden nodes was kept fixed at 12 nodes in both cases. It was found that in most cases the best ANN performance without GA optimization was achieved with 12 hidden nodes. The size of standalone ANNs is significantly larger with several input nodes than the GA supported ANNs having only ten inputs. In the next comparison, the number of hidden nodes is also evolved with GA, and thus the size of ANN further changed every time a new chromosome is obtained. Each ANN is trained for 50-100 epochs before using the test data to measure its performance. To limit the computational time to reasonable extents, the stopping criterion was preset to the number of epochs, since each generation required training 200 ANNs, and in each case the GA is allowed to evolve over 100 generations. A resilient back-propagation training algorithm has been shown to work faster on larger networks by Riedmiller and Braun (1993), and has been used throughout this study. A batch training strategy is employed with *tan-sigmoid* activation functions for the input and hidden layers and a pure-linear activation function for the output layer.

Results and Discussion

Experiments were carried out with constant crossover and mutation parameters for result comparisons between evolved ANN and fixed sized ANN. All results have been summarized in Tables 1 and 2. The numbers in the first column represent the data set. The next columns describe the training conditions, represented by alphabets explained below. In table 2 an extra column shows the number of hidden nodes (*n*), GA converged to. Figure 4 shows the resulting convergence curves for both fitness functions.

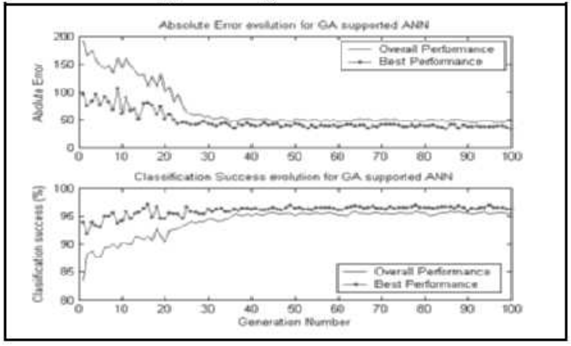


Fig. 4. GA convergence curves for the two fitness functions.

It can be seen that in all cases GA supported ANNs clearly outperform the standalone ANNs. In some cases; for instance when using features on Sum signal (dataset 2), the ANN completely failed because the feature values were zero in many cases due to very small low-frequency content change in the signal. That in turn diminishes the statistical features computed based on it. GA, however uses only the non-zero features to train the ANN and thereby showing a significantly better performance. In case 5 (including case 2 data) ANNs do not train very well, and once again GA-ANNs result in superior performance. Table 2 compares the performance of standalone ANNs with GA evolved ANNs (with optimized number of hidden nodes). The number of hidden nodes converged to 7 in most cases. For the dataset used in this study, it can be seen that statistical features give the best performance, with minimum absolute errors and maximum classification success. However, when all features are used together (dataset 5) to search for the best solution, the performance further improves, and this could be a direct result of the fact that good features from other datasets give the best performance in conjunction with good statistical features. Since the training times for all these different cases were too high, only 50 epochs were used in most cases, sufficient to show the resulting improvement upon using the GA. Overall, it can be concluded that GA-evolved ANNs perform much better, and can lead to considerable savings of computational expenses for effective bearing health monitoring. Therefore, once these optimal parameters have been found, such a system can be used for online classification of the faults. In that case we expect to use a select feature set and a fixed ANN structure to identify and classify the fault. Such optimal parameters can be scheduled for re-computation to take into account the non-stationary nature of the process. These kinds of calibrations can be done concurrently offline and employed online once the new parameters are available.

Table 1. Comparing performance of standalone ANNs with GA supported ANN

Training Conditions					Min Absolute Error		Mean Absolute Error		Best Classification Success (%)		Mean Classification Success (%)	
D[†]	E/G/P/C/M *				ANN-12	GANN-12	ANN-12	GANN-12	ANN-12	GANN-12	ANN-12	GANN-12
1	100	100	20	5/3	84.90	31.9	131	47.9	97	100	92	98.2
2	75	100	20	5/1	815	32.6	816	62.2	22.2	88	22	82
3	50	100	20	5/3	97.85	64.99	115.3	94.79	98.5	99.67	97.2	97.6
4	50	100	20	5/3	82.11	57.27	601.1	114.45	95.1	98.1	65.22	95.5
5	50	100	20	5/3	2048	18.42	2048	39.63	22	100	22	99.65

Table 2. Comparing the performance of fixed size ANNs with GA evolved ANNs.

Training Conditions			Min Absolute Error		Mean Absolute Error		Best Classification Success (%)		Mean Classification Success (%)	
D[†]	E/G/P/C/M*	n	ANN-12	GANN-n	ANN-12	GANN-n	ANN-12	GANN-n	ANN-12	GANN-n
1	50/50/20/5/5	7	84.90	12.12	131	22.53	97	100	92	99.46
2	50/50/20/5/5	7	815.00	111.8	816	155.8	23	99.22	22	96.58
3	50/50/20/5/5	8	97.85	45.76	115.3	87.12	98.5	100	97.2	99.3
4	50/50/20/5/5	8	82.11	91.35	601.1	115.16	95.1	95.49	65.22	93.5
5	50/50/20/5/5	7	2048	11.9	2048	23	22	100	22	99.94

[†] **1**: Statistical Features (38), **2**: Sum features (38), **3**: Difference features (38), **4**: Spectral Features (128), **5**: All the features (242) * D: Data Set, E: # epochs for ANN training, G: Number of generations, P: Population size, C: crossover points, M: mutation points, *n*: Number of hidden node

Conclusion and Future Work

It has been shown that using GAs for selecting an optimal feature set for a classification application of ANNs is a very powerful technique. Irrespective of the vastness of the search space, GA can successfully identify a required number of good features. The definition of ‘good” is usually specific to each classification technique, and GA optimizes the best combination based on the performance obtained directly from the success of the classifier. The success of the ANNs in this study was measured in both terms; i.e., the training accuracy and the classification success. Although the end result depends on the classification success, the training accuracy is the key element to ensure a good classification success. Having achieved that objective, further scope of this research is in two directions. First, to make the ANN structure more adaptive so that not only the number of hidden nodes but also the number of input nodes and the connection matrix of the ANNs can be evolved using GA. Second, to use this technique to find out a desired set of good features for more complex systems such as planetary gear systems, where some faults cannot be easily distinguished.

Acknowledgments

The authors would like to thank the Woodroof School of Mechanical Engineering, and the Intelligent Control Systems Laboratory at the School of Electrical and Computer Engineering, the Georgia Institute of Technology, for the experimental data sets obtained for the work.

References

Billington S A (1997), ‘Sensor and Machine Condition Effects in Roller Bearing Diagnostics”, *Master’s Thesis, Department of Mechanical Engineering*, Georgia Institute of Technology, Atlanta.

Balakrishnan K, Honavar V (1995), ‘Evolutionary Design of Neural Architecture – A Preliminary Taxonomy and guide to Literature”, *Artificial Intelligence Research group, Iowa State University, CS Technical Report #95-01.*

Drakos N, ‘Genetic Algorithms as a Computational Tool for Design” http://www.cs.unr.edu/~sushil/papers/thesis/thesishtml/thesishtml.html

Dallal G E (2004), ‘The Little Handbook of Statistical Practice”, http://www.tufts.edu/~gdallal/LHSP.HTM.

Duda R O, Hart P E, Stork D G (2001), *Pattern Classification* , Second Edition, Wiley-Interscience Publications.

Edwards D, Brown K, Taylor N(2002), ‘An Evolutionary Method for the Design of Generic Neural Networks”, CEC '02. *Proceedings of the 2002 Congress on Evolutionary Computation*, vol. 2 , pp. 1769 –1774.

Filho E F M, de Carvalho A (1997), 'Evolutionary Design of MLP Neural Network Architectures", *Proceedings of IVth Brazilian Symposium on Neural Networks*, pp. 58 - 65

Goldberg D E (1989), *Genetic Algorithm in Search, Optimization and Machine Learning*, Addison Wesley Publishing Company

Holland J (1975), *Adaptation In Natural and Artificial Systems*. The University of Michigan Press, Ann Arbor

Jack L B, Nandi A K (2000), 'Genetic Algorithms for Feature Selection in Machine Condition Monitoring With vibration Signals", *IEE Proceedings, Image Signal Processing*, Vol. 147(3), June, pp 205-212.

Lawrence S, Giles C L, Tsoi A C (1997), 'Lessons in Neural Network Training: Overfitting May be Harder than Expected", *Proceedings of the Fourth National Conference on Artificial Intelligence*, AAAI-97, pp 540-545.

Miguel ÁCarreira-Perpiñán (1996), *A Review of Dimension Reduction Techniques* (1997), Dept. of Computer Science, University of Sheffield, technical report CS-96-09.

Riedmiller M, Braun H (1993), "A direct adaptive method for faster backpropagation learning: the RPROP algorithm", *IEEE International Conference on Neural Networks*, vol. 1, pp. 586 –591.

Samanta B (2004a), 'Gear Fault Detection Using Artificial Neural Networks and Support Vector Machines with Genetic Algorithms", *Mechanical Systems and Signal Processing*, Vol. 18, pp. 625-644.

Samanta B (2004b), "Artificial Neural Networks and Genetic Algorithms for Gear Fault Detection", *Mechanical Systems and Signal Processing* (Article in press).

Shiroishi J, LiY, Liang S, Kurfess T, Danyluk S (1997), 'Bearing Condition Diagnostics via Vibration & Acoustic Emission Measurements", *Mechanical Systems and Signal Processing*, 11(5

Sunghwan S, Dagli C H (2003), Proceedings of the International Joint Conference on Neural Networks, vol. 4, pp. 3218 –3222.

Tang K S, Man K F, Kwong S, He Q (1996), 'Genetic Algorithms and Their Applications", *IEEE Signal Processing Magazine* (11), pp 21-37

Vlachos M, Domeniconi C, Gunopulos G, Kollios G (2002), 'Non-Linear Dimensionality Reduction Techniques for Classification and Visualization", *Proceedings of 8th SIGKDD* Edmonton, Canada.

The Applications of Soft Computing in Embedded Medical Advisory Systems for Pervasive Health Monitoring

Bing-Nan LI; Ming-Chui Dong; Vai Mang I

Dept. of Electrical and Electronics Engineering, University of Macau, Macau S.A.R., China;
INstituto de Engenharia de Sistemas e Computadores de Macau, Macau

Abstract. Based on the idea of SoC (System-on-Chip), an embedded-link, self-adaptive medical advisory system is proposed for home health monitoring, which can be merged into the embedded platform of mobile or wearable medical sensors and run independently. Meanwhile, if necessary, it will up link to health centers and self-calibrate its knowledge base. To provide effective medical advices, the methods of soft computing, including temporal fuzzy variables and weighted medical rules, are applied to the proposed medical advisory system. The elementary components, including medical knowledge base, inference machine and shell, are addressed in detail. Lastly, the way to construct such embedded-link, self-adaptive medical advisory system and integrate it into a mobile cardiovascular monitoring device is introduced too.

Keywords: medical advisory system, temporal fuzzy, weighted rules, health monitoring

1 Introduction

Nowadays, people pay more and more attention to their health condition. However, the frequent visiting to hospitals, mostly for routine health examination and/or medical consultation, is a heavy load for patients as well as for physicians and nurses. Hereby the research on home healthcare, especially home healthcare monitoring, is booming due to improving the quality of people’s life while depressing medical expenditure. The reported research works focus on constructing the platform for home health monitoring. It collects the vital signs (Franchi et al. 1998; Jannett et al. 2002) or monitors users’ daily life (Ogawa et al. 2002; Castro et al. 2001), then submits the collected data to central server for further medical advices via various communicating mechanisms (Nambu et al. 2000; Crumley et al. 2000).

Recently, with the prevalence of mobile apparatus and embedded systems, the research on wearable medical sensors and pervasive health monitoring is increasing popular (Diana et al. 2003). However, in terms of these systems, one of the common flaws is deficiency of on-system intelligence. For example, most of them just generate alarming messages if and only if there are some abnormal symptoms. But from the viewpoint of medicine, it is generally too late for the diseases might have become unrecoverable once the signs were abnormal. Therefore, to promote the pervasive health monitoring, it is essential to develop the home users-oriented medical advisory systems suitable for the embedded platforms. The desired system will analyze the long-term symptoms to assess users' health condition or find out the intrinsic information about the diseases' evolution.

In this paper, an embedded-link, self-adaptive medical advisory system is proposed, which can be merged into the embedded platform of mobile or wearable medical sensors and run independently. Meanwhile, if necessary, it will up link to health centers and self-calibrate its knowledge base. Since it is designed for home users with little of medical professional knowledge, the medical advisory system will be built on the legible linguistic framework. The methods of soft computing, including temporal fuzzy variables and weighted medical rules, are applied to the proposed system.

2 Building the Medical Knowledge Base

Most of the reported medical expert systems seem too professional to be applied in home healthcare. For example, the diagnostic program of heart diseases at MIT demands more than 100 vital parameters including the abstruse data from laboratory testing. It is useful to aid medical professionals at health centers but doomed to fail in home healthcare. As to home users with little medical knowledge, for home health monitoring, they merely care the evolution of health condition, and if any, the related medical advices in daily life. Benefited from the advanced technology of machine learning, it is feasible to elicit the medical advisory knowledge from the long-term monitoring dataset of diagnosed patients. Different from the knowledge for medical diagnosis, which is defined and described rigorously, here the elicited knowledge base comprises a series of medical rules with temporal fuzzy variables.

2.1 Temporal Fuzzy Variables

Generally, the health monitoring devices collect several vital signs so to reflect the health condition of home users. Moreover, for mobility, these home-oriented devices only conserve limited times of monitoring data. Therefore, it is essential to introduce a temporal variable to reflect the long-time variation of vital signs.

Suppose the monitored vital signs are symbolized as "VS_1, VS_2… VS_n" and for each of them, in terms of the traditional mobile health monitoring devices, merely limited amount of historical data are recorded as "VS_{i1}, VS_{i2}…VS_{im}" ($i \in [1, n]$) according to the time of acquisition. To reflect the accumulative medical information, a pair of temporal variables "VS_i, k_i"is introduced for medical advisory system. Here "VS_i", defined by equation (1), synthesizes the historical records and 'k_i', defined by equation (2), reflects the variation.

$$VS_i = \begin{cases} VS_{i1} & \textit{Initializtion} \\ \dfrac{VS_i + VS_i^{new}}{2} & \textit{Updating} \end{cases} \tag{1}$$

where " VS_i^{new} " means the newest monitoring record such as "VS_{i2}", "VS_{i3}"… It is shown that "VS_i" eventually updates as following:

$$VS_i = VS_{i1} \to \frac{VS_{i1} + VS_{i2}}{2} \to \frac{VS_{i1} + VS_{i2} + 2VS_{i3}}{2^2} \to \frac{VS_{i1} + VS_{i2} + 2VS_{i3} + 2^2 VS_{i4}}{2^3} \cdots \\ \to \frac{VS_{i1} + VS_{i2} + \sum_{\mathrm{j=3}}^{\mathrm{m}} 2^{(j-2)} VS_{\mathrm{ij}}}{2^{(\mathrm{m}-1)}} \to \cdots \tag{2}$$

Hence, though quite simple computation in equation (1), the variable "VS_i" has been implicitly weighted according to its temporal sequence.

$$k_i = \begin{cases} 0 & \textit{Initialization} \\ Sign|v_i| \left| \dfrac{|v_i| - |v_i^{old}|}{|v_i| + |v_i^{old}|} \right| & \textit{Updating} \end{cases} \tag{3}$$

where "$|\ |$"is to extract the absolute value while "$Sign|\ |$" is to get the sign. The values "v_i" and "v_i^{old}" are respectively defined as follows.

$$v_i = \begin{cases} 0 & \mathit{Initialization} \\ \left.\dfrac{v_i + \dfrac{\Delta VS_i}{\Delta T}}{2} = \begin{cases} \dfrac{v_i + (VS_i^{new} - VS_i)}{2} & \Delta T = 0 \\ \dfrac{v_i + (VS_i^{new} - VS_i)}{2\Delta T} & \mathit{Otherwise} \end{cases}\right\} & \mathit{Updating} \end{cases} \tag{4}$$

And "v_i^{old}" is evaluated as the "v_i" before updating. Here "ΔT" comes from the day-based monitoring interval.

In equation (3), "k_i"has been generalized into the interval "$(-1, 1)$"so it is appropriate to fuzzify the variable "k_i"for later fuzzy medical advisory system. According to Figure 1, the correspondent membership function of temporal fuzzy variable "μ_i" is defined via the following equations (Pedrycz, 1998):

$$\begin{aligned}
&\mu_i^{VU} = \begin{cases} 1, & x \in [0.8, 1] \\ (x-0.6)/0.2, & x \in [0.6, 0.8) \end{cases}
&&\mu_i^{MU} = \begin{cases} (0.7-x)/0.2, & x \in [0.5, 0.7] \\ (x-0.3)/0.2, & x \in [0.3, 0.5) \end{cases} \\
&\mu_i^{U} = \begin{cases} (0.4-x)/0.2, & x \in [0.2, 0.4] \\ x/0.2, & x \in [0, 0.2) \end{cases}
&&\mu_i^{N} = \begin{cases} (0.2-x)/0.2, & x \in [0, 0.2] \\ (x+0.2)/0.2, & x \in [-0.2, 0) \end{cases} \\
&\mu_i^{D} = \begin{cases} -x/0.2, & x \in [-0.2, -0] \\ (0.4+x)/0.2, & x \in [-0.4, -0.2) \end{cases}
&&\mu_i^{MD} = \begin{cases} (-0.3-x)/0.2, & x \in [-0.5, -0.3] \\ (x+0.7)/0.2, & x \in [-0.7, -0.5) \end{cases} \\
&\mu_i^{VD} = \begin{cases} (-0.6-x)/0.2, & x \in [-0.8, -0.6] \\ 1, & x \in [-1, -0.8) \end{cases}
\end{aligned} \tag{5}$$

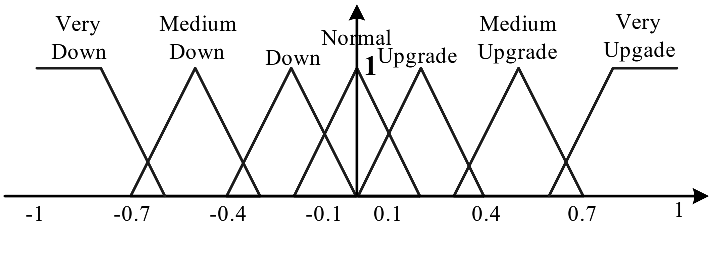


Figure 1. Fuzzifying the temporal variable

2.2 Weighted Medical Rules

To implement the uncertain medical inference and diagnosis, besides the preceding methodology of temporal fuzzy variables, most of the medical rules are also assessed via the certainty factors as well (Giarratano, 1998). Consequently, the classical medical rules could be described as following:

$$R_t : \{\wedge[(Sign, Attribute)_i \mid Hypothesis_i]\} \Rightarrow \{\wedge(Disease \mid Hypothesis)_j\}\ CF_t \tag{6}$$

where the subscript '*t*' means the *t*-th medical rule; '*i*' is the *i*-th antecedent and similarly, '*j*' is the *j*-th consequent of the specific medical rule. Moreover, "*Sign*" belongs to the collected vital signs for health assessing and its attribute is the preceding fuzzy variable such as "*Very Upgrade*", "*Medium Upgrade*", etc. And the symbol '^' stands for the logic "*AND*" while '|' means the alternative item. Finally, "*CF*" is the certainty factor grading the specific medical rule.

As to the medical diagnosis, in general, it commences with the peripheral factors such as medical history, appearance symptoms and so on. Hereby medical professionals could derive several possible diseases from the known medical information. Then, the final diagnosis will only be concluded from the object-oriented measuring or testing. Consequently, in terms of the medical rules, the symptoms or signs in antecedents play the different contributive roles for the consequents. To improve the uncertain diagnosis, it is natural to weight the antecedents of medical rules.

Here the methodology based on statistical average and generalization is proposed to weight the antecedents. For the specific *i*-th antecedent "$(Sign, Attribute)_i$" or "$Hypothesis_i$", if it exists in *n* piece of medical rules, the corresponding weight could be worked out as follows.

$$weight_i = (\sum_{k=1}^{n} CF_{nk}) / n \tag{7}$$

Then, for each of medical rule, the weights of antecedents could be further generalized as (8). For example, if the *t*-th medical rule comprises *p* antecedents, the *k*-th weight:

$$weight^*_{tk} = weight_{tk} / \sum_{w=1}^{p} weight_{tw} \quad (k \in [1, p]) \tag{8}$$

Finally, the mentioned rules as (6) will be converted into the medical rules compliant to (9):

$$R_t : \{\wedge[(Sign, Attribute)_i \mid Hypothesis_i, weight_i]\} \Rightarrow \{\wedge(Disease \mid Hypothesis)_j\}\ CF_t \tag{9}$$

3 Embedded Medical Advisory Systems

As mentioned previously, the embedded-link, self-adaptive medical advisory system will be integrated with the medical sensors for pervasive health monitoring. Here the "embedded-link" system means it should be well designed and ready for the embedded platforms of mobile or wearable medical sensors. Linking to central server enables it to provide the effective health services. And the "self-adaptive" system refers to running independently for the various medical sensors and home users due to the customized and updatable diagnostic knowledge base.

3.1 Accompanied Knowledge Base

Since the module of machine learning for medical knowledge discovery is generally built on the tremendous dataset and moreover, depends on the support of medical experts, it is unfeasible and also not necessary to implement the complicated module in the embedded platform. The embedded medical advisory system hereby comprises the module of inference machine and a customized & updatable knowledge base. Here "customized" refers to only putting those essential medical rules related to the specific medical sensor and health condition in local knowledge base. Via up linking to the global comprehensive knowledge base residing in health centers, each of the individual embedded medical advisory system could be calibrated conveniently while necessary.

During the initialization, the relevant medical information is firstly submitted to health centers. Assume that the submitted information could be formatted as "$\{\ \wedge(Entity, Value\,|\,Attribute)_q\ \}$", where the "$Entity$" belongs to either the collected vital signs or the basic physiological & physical parameters of home user. Moreover, as mentioned previously, the medical rules in global knowledge base are compliant to (9). Therefore, to customize the local knowledge base is to search & match the most suitable medical rules according to the submitted information, and then download them to the local accompanied knowledge bases. This customizing and updating procedure can be conducted dynamically during system initialization or data backup.

3.2 Inference Machine

Given the uncertain medical rules with temporal fuzzy variables, the classical fuzzy inference is built on the "$conjunction - disjunction$" operation (Yager, 1992). Now assume that the final diagnosis could be specified as "*CR* (*Disease$_j$*)", where "*CR*" is the acronym of "*Confidence of Result*" corresponding to the "*CF*: *Certainty Factor*". Then, for the medical rule compliant to (6), the fuzzy inference formula could be summarized as (10):

$$CR(Disease_j) = \{\vee[CF_t * \{\wedge\mu[(Sign, Atribute)_i \mid Hypothesis_i)]_t\}]\} \tag{10}$$

where ‘ $\vee$ ’ is the operator of “*disjunction*” and ‘ $\wedge$ ’ is “*conjunction*”.

However, due to weighted, here the uncertainty comes from not only the fuzzy variables but also the weights of medical rules. Consequently, in the proposed embedded medical advisory system, the inference machine should be modified accordingly. Similarly, the first is also an accumulative procedure because of the incomplete matching. In (11), the involved “*hypotheses*” should be firstly worked out for following medical diagnosis. Then the modified formula in (12) acts as the inference foundation of medical advisory system.

$$CR(Hypothesis_l) = \{\vee[CF_m * \sum_p \mu(vs_{mp}) * weight_{mp}]\} \tag{11}$$

$$CR(Disease_k) = \{\vee[CF_n * \sum_q \mu(vs_{nq}) * (weight_{nq} * [1 \mid CR_l])]\} \tag{12}$$

3.3 Shell

In the knowledge-based system, “shell” means that the inference machine is separated from the knowledge base so to run adaptively for the different applications. Here due to the limited systematic resources, as mentioned previously, there are only the customized medical rules in the accompanied knowledge base. Naturally, the embedded medical advisory system needs to update its knowledge base while necessary. Benefited from the distributed framework of network communication, the local knowledge base is convenient to be updated and modified.

Eventually, to implement the pervasive health monitoring, the accompanied knowledge base should be dynamically updated so that the embedded medical advisory system keeps effective. For example, once the collected symptoms are out of the boundary of knowledge base, the embedded medical advisory system will report the related information to health centers. If it is under the control of global medical knowledge base, the inference results accompanied with the updated knowledge base will be fed back accordingly. Otherwise, the request will be transferred to medical professionals. Not only the medical diagnosis is sent back, but also the accompanied medical knowledge base will be reviewed and revised by medical professionals. Moreover, due to the ongoing machine learning for medical knowledge discovery, the embedded medical advisory system will be regularly calibrated too.

4 Prototyping System

Presently, the proposed medical advisory system has been incorporated into a mobile cardiovascular monitoring system based on the hemodynamic analysis of sphygmogram. It firstly works out a group of vital signs from the sampled sphygmogram, which reflects the health profile of cardiovascular system including the cardiac and microcirculation function, vascular and blood status, etc. Therefore, besides conserved for later reference, these vital signs are also submitted to the embedded medical advisory system. In virtue of the proposed embedded-link self-adaptive medical advisory system, if the effective medical advices are provided, they will be fed back to home users directly. Otherwise, the collected medical information will be submitted to health center and asks for the medical assistance.

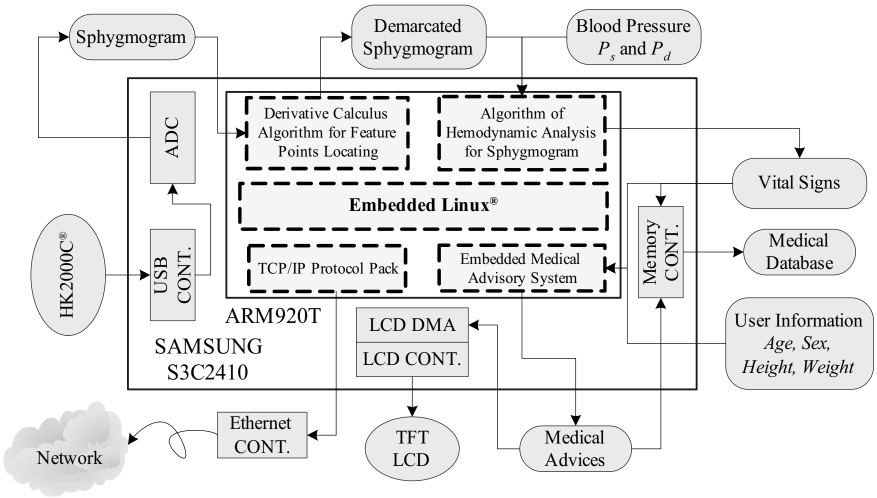


Figure 2. The ARM-based intelligent mobile cardiovascular monitoring system

Here the medical sensor – ‘HK2000C ®” (*By Hefei Huake Electron Graduate School, China*) –is adopted for sampling of sphygmogram. The collected data are then submitted to a self-contained developing platform of embedded systems –‘HHARM2410 ®” (*By HHTech Co. Ltd., China*) – via the integrated USB port. The following signal processing and data analyzing, coordinated by the on-system ‘embedded Linux ®”, is condensed into a micro system, shown in Figure 3, with the sophisticated ARM9 chip ‘Samsung ® S3C2410”, 64M SDRAM and 16M Flash ROM, etc. Then, with the extended peripheral interfaces such as ‘Ethernet interface” and ‘TFT LCD”, the mobile device with embedded medical advisory system could intelligently monitor and analyze the health condition of cardiovascular system via sphygmogram.

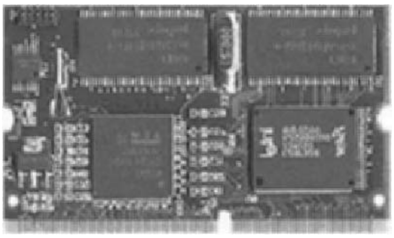


Figure 3. The core module of prototyping system

5 Conclusion

In this paper, an embedded-link, self-adaptive medical advisory system is proposed for pervasive health monitoring. The system is well designed and implemented for the embedded platform of mobile or wearable medical sensors. Moreover, to conduct the uncertainty medical diagnosis, the methodologies of soft computing, such as temporal fuzzy variables and weighted medical rules, are applied to the proposed medical advisory system. Finally, it is implemented and verified in a mobile cardiovascular monitoring system.

In the future, the proposed system could be further enhanced. For example, though the weighted medical rules with temporal fuzzy variables are effective to deal with the uncertainty medical diagnosis, the newer methodologies of soft computing such as fuzzy neural networks will be more competitive for the embedded systems. And it is also attractive to implement the proposed system for the autonomous interpreter of electrocardiogram.

Acknowledgments

The authors would like to acknowledge the financial support of Research Committee of University of Macau (Grant No: 3676) in our research project ‘Network-based, Intelligent, Home Healthcare System’.

References

D. Castro, J. Presedo, M. Fernandez-Delgado, and S. Barro (2001) Patient telemonitoring at home. In: Proceedings of 23rd Annual EMBS International Conference, pp. 3571-3574.

Diana M., Radu M., Wungmee P., and Sundaresan J. (2003) Ready to Ware. IEEE Spectrum, pp. 28-32.

D. Franchi, A. Belardinelli, G. Palagi, A. Ripoli, and R. Bedini (1998) New telemedicine approach to the dynamic ECG and other biological signals ambulatory monitoring. Computers in Cardiology 25: 213-215.

G. C. Crumley, N. E. Evans, W. G. Scanlon, J. B. Burns, and T. G. Trouton (2000) The design and performance of a 2.5-GHz telecommand link for wireless biomedical monitoring. IEEE Trans. Inform. Technol. Biomed. 4(4): 285-291.

J. Giarratano, and G. Riley (1998) Inexact reasoning. In: Expert Systems Principles and Programming. PWS Publishing Company, pp. 227-308.

M. Nambu, K. Nakajima, A. Kawarada, and T. Tamura (2000) A system to monitor elderly people remotely using the power line network. In: Proceedings of 22nd Annual EMBS International Conference (1): 782-785.

M. Ogawa, R. Suzuki, and S. Otake (2002) Long term remote behavioral monitoring of elderly by using sensors installed in ordinary house. In: Proceedings of 2nd Annual International IEEE-EMBS Special Topic Conference on Microtechnologies, Medical and Biology, pp. 322-325.

R.R. Yager, and L.A. Zadeh (1992) Expert systems using fuzzy logic. In: An Introduction to Fuzzy Logic Applications in Intelligent Systems. Kluwer Academic Publishers, Norwell, Massachusetts.

T. C. Jannett, S. Prashanth, S. Mishra, and V. Ved (2002) An intelligent telemedicine system for remote spirometric monitoring. In: Proceedings of 34th Southeastern Symposium on System Theory, pp. 53-56.

W. Pedrycz, and F. Gomide (1998) Linguistic Variables. In: An Introduction to Fuzzy Sets. The MIT Press, Cambridge, Massachusetts.

Application of Fuzzy Inference Techniques to FMEA

Kai Meng Tay and Chee Peng Lim*

School of Electrical and Electronic Engineering, University of Science Malaysia

Engineering Campus, 14300 Nibong Tebal, Penang, Malaysia
Email (*corresponding author): cplim@eng.usm.my

Abstract. In traditional Failure Mode and Effect Analysis (FMEA), the Risk Priority Number (RPN) ranking system is used to evaluate the risk level of failures, to rank failures, and to prioritize actions. This approach is simple but it suffers from several weaknesses. In an attempt to overcome the weaknesses associated with the traditional RPN ranking system, several fuzzy inference techniques for RPN determination are investigated in this paper. A generic Fuzzy RPN approach is described, and its performance is evaluated using a case study relating to a semiconductor manufacturing process. In addition, enhancements for the fuzzy RPN approach are proposed by refining the weights of the fuzzy production rules.

Keywords: FMEA, fuzzy logic, Fuzzy Production Rules, Weighted Fuzzy Production Rules

1. Introduction

Failure Mode and Effect Analysis (FMEA) can be described as a systemized group of activities intended to recognize and to evaluate the potential failures of a product or a process and its effects. FMEA identifies actions which could eliminate or reduce the chance of the potential failure from recurring (Chrysler Corporation et. al., 1995). It also aids in identifying the key design or process characteristics that require special controls for manufacturing, and in highlighting areas for improvement in process control (Ireson et al. 1995).

Over the years, investigations have been conducted to enhance the FMEA methodology using Artificial Intelligence (AI) techniques. Fuzzy Cognitive Map was applied to FMEA to allow symbolic reasoning instead of numerical reasoning in FMEA (Peláez et al. 1996). Fuzzy inference technique has been applied to improve the failure risk evaluation, ranking and prioritization abilities of FMEA (Bowles et al. 1995). Use of Fuzzy Reasoning and Grey Relation with FMEA for marine industry has been proposed to address the traditional FMEA weaknesses in failures risk evaluation (Pillay et al. 2001). A pure fuzzy logic system was applied to FMEA of an auxiliary feed water system in a nuclear power plant for failure risk ranking enhancement (Guimarães et al. 2004). Besides, a fuzzy assessment of FMEA for engine system had been reported (Xu et al. 2002).

This paper focuses on the use of fuzzy inference techniques as an alternative to overcome the shortcomings associated with the traditional FMEA risk ranking system. The traditional FMEA uses the Risk Priority Number (RPN) ranking system to evaluate the risk level of failures, to rank failures, and to prioritize actions (Chrysler Corporation et al. 1995). By comparing with the traditional FMEA, the fuzzy approach allows failure risk to be evaluated and prioritized based on experts' knowledge. A generic approach for Fuzzy RPN function modeling is introduced in this paper. In addition, we propose to enhance the Fuzzy RPN model by refining the global weights of the fuzzy production rules. Performance of the proposed approach is evaluated using a case study relating to a semiconductor manufacturing environment.

2. Failure Risk Evaluation, Ranking, and Prioritization Issues in FMEA

In FMEA, the RPN is determined by multiplication of three input factors, i.e., Severity, Occurrence, and Detect, as follows:

$$RPN = Severity \times Occurrence \times Detect \qquad (1)$$

Severity is an assessment of the effect of potential failure mode to the next component, subsystem, or customer. Occurrence is defined as the likelihood that a specific cause will occur. Detect is an assessment of the ability of current design control to detect a potential cause (Chrysler Corporation et al. 1995). In general, these three factors are estimated by FMEA users using a scoring system from "1" to "10" based on a commonly agreed evaluation criteria. As the RPN is a measure of failures risk, it is used to rank failures and to prioritize actions. Actions will then be taken with priority given to the failure with the highest RPN.

The traditional RPN ranking has been well accepted for safety analysis in FMEA. However, it suffers from several weaknesses. It has been pointed out that the same RPN value can be obtained from a number of different score combinations of Severity, Occurrence, and Detect, but the risk can be different (Ben-Daya et al. 1993, Pillay et al. 2003). Besides, the relative significance of the three factors is neglected in the traditional RPN calculation. These three factors are assumed to be of equal importance, but this may not be the case in practice. Indeed, researches have shown that the relative significance of the three factors varied based on the nature of a process or a product.

3. Fuzzy Production Rules and Weighted Fuzzy Production Rules

In a Fuzzy Inference System (FIS), experts' knowledge is represented with a rule base comprising fuzzy production rules (FPRs) in the If-Then format. All the FPRs consist of two parts: antecedents/inputs and consequents/outputs. A FIS uses the FPRs to determine mappings from input universe of discourse to output universe of discourse (Jang et al. 1997, Lin et al. 1995). The non-weighted FPRs assume all the rules are certain and have equal weights. Besides, the FPRs consider that all the antecedents are mapped perfectly (100%) to the consequents. However, this may not be true all the time.

Weighted Fuzzy Production Rules (WFPRs) represents an enhancement for FPRs. WFPRs allow knowledge imprecision to be taken into account by adding extra knowledge representation parameters such as threshold value, certainty factor, local weight, and global weight (Yeung et al. 1997). The global weight concept is used to represent the relative degree of importance of the contribution of each rule towards the output membership function. There are two different applications of the global weight concept in the FIS, i.e., (i) to compare the relative degree of importance of rule with each of the rules in a given inference path leading to a specific output membership function; and (ii) to show the relative degree of importance of the same rule when it is used in different inference paths leading to different output membership functions. From (ii), the global weight allows an antecedent to be mapped to several consequents with different weights. It has the ability to handle the condition that an antecedent is not mapped perfectly to a consequent. If an antecedent is relatively more important to consequent X than to consequent Y, then the rule will have a higher global weight assignment associated with X than with Y.

4. A Generic Modeling Approach to the Fuzzy RPN Function

4.1 Fuzzy Membership Functions

Severity, Occurrence, and Detect, as use in the traditional RPN function, are used as the input factors for the Fuzzy RPN function. The membership functions of the three factors are determined by interpreting the linguistic terms. Tables 1, 2, and 3 summarize some classifications/criteria describing each linguistic term.

Table 1. Scale table for Severity

Severity Scale table		
Rank	Linguistic Term /Classification	Criteria
10	Very High (Liability)	Failure will affect safety or compliance to law.
9~8	High (Reliability / reputation)	Customer impact. Major reliability excursions.
7~6	Moderate (Quality / Convenience)	Impacts customer yield. Wrong package/par/marking. Customer special handling/ equipment damage.
5~2	Low (Internal Yield / Special Handling)	Yield hit, Cosmetic.
1	None (Unnoticed)	Unnoticed either internally or externally.

Table 2. Scale table for Occurrence

Occurrence scale table		
Rank	Linguistic Term/Classification	Criteria
10~9	Very High	Many/shift, Many/day
8~7	High	Many/week, Few/week
6~4	Moderate	Once/week, Several/month, Few/quarter
3	Low	Once/month
2	Very Low	Once/quarter
1	Remote	Once ever

Table 3. Scale table for Detect

Detect scale table		
Rank	Linguistic Term/Classification	Criteria
10	Extremely Low	No Control available.
9	Very Low	Controls probably will not Detect
8~7	Low	Controls may not Detect excursion until reach next functional area.
6~5	Moderate	Controls are able to Detect within the same functional area
4~3	High	Controls are able to Detect within the same machine/module.
2~1	Very High	Controls will Detect excursions before next lot is produced. Prevent excursion from occurring

In this paper, Gaussian membership function is used to represent each input, as follows.

$$Gaussian(x;c,\sigma) = e^{-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2} \qquad (2)$$

where c represents the membership function center and represents the membership function width. For every input, c is fixed to "1" for the first linguistic term; "10" for the last linguistic term; and for others the center of each term. Parameter   is tuned so that every membership function has about 50 % overlapping. This will eliminate the risk of introducing an "empty pole" in the inputs domain (Jang et al. 1997). Figures 1, 2, and 3 depict the membership functions for Severity, Occurrence, Detect, respectively. Output of the Fuzzy RPN model, i.e., the RPN value, is divided into five equal partitions, namely Low, Low Medium, Medium, High Medium, and High, as shown in Figure 4.

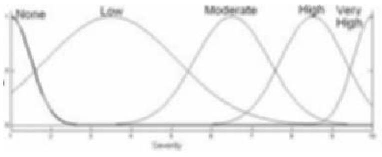


Fig. 1. Membership function for "Severity"

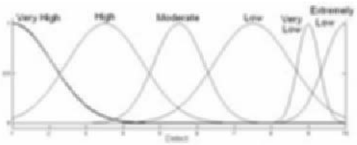


Fig. 3. Membership function for "Detect"

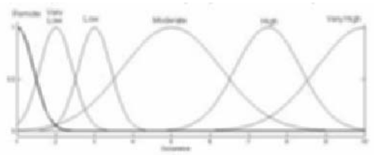


Fig. 2. Membership function for "Occurrence"

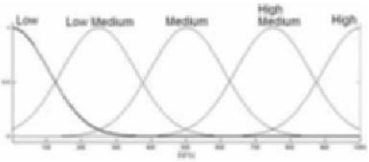


Fig. 4. Membership function for the output RPN value.

4.2 Fuzzy Rule Base

Fuzzy rule base is a collection of experts' knowledge in the FRR or WFPR representation. In the above case, the fuzzy rule base will have 180 (5 (Severity) x 6 (Occurrence) x 6 (Detect)) rules. As an example, both the antecedents can be mapped to output term *High* in the FPR representation, as follows.

Rule 1

IF Severity is **Very high** and Occurrence is **Very High** and Detect is **Extremely Low,** THEN RPN is **High**.

Rule 2

IF Severity is **Very high** and Occurrence is **Very High** and Detect is **Very Low,** THEN RPN is **High**.

Instead of FPR, we use WFPR in this paper. The above two rules can be refined by changing the global weights associated with each rule. For example, based on the experts' opinion, Rule 1 is mapped to High with 100% while Rule 2 is mapped to High with 95 % and High Medium with 5%. Using this method, the mapped outputs of the two antecedents can be differentiated and represented more effectively without increasing the number of the output term.

Rule 1

IF Severity is **Very high** and Occurrence is **Very High** and Detect is **Extremely Low**, THEN RPN is **High**. (weight 1.00)

Rule 2

IF Severity is **Very high** and Occurrence is **Very High** and Detect is **Very Low**, THEN RPN is **High** (weight 0.95)

IF Severity is **Very high** and Occurrence is **Very High** and Detect is **Very Low**, THEN RPN is **High Medium**. (weight 0.05)

4.3 Properties of the Proposed Fuzzy RPN Model

4.3.1 Monotone output

Output of the RPN function, i.e., the RPN value, is a measure of failure risk. Ideally, the higher the score of the RPN value, the more critical the situation of a failure is. From the inputs-output relationship, it can be concluded that the RPN value should increase as any one of the three input scores increases. This property will ensure the RPN model to give logical risk evaluation and enable failures to be compared more effectively. For example, for two failures with input sets 5 5 5 and 5 5 6 (Severity, Occurrence, and Detect, respectively), the RPN value for the second failure should be higher than that of the first. Prediction is deemed illogical if the RPN model yields a contradictory result.

4.3.2 Output resolution (Sensitivity of output to the changes of input)

Resolution refers to the sensitivity of the Fuzzy RPN model in giving a representative output with respect to the changes of the inputs. For example, in the previous example with input sets 5 5 5 and 5 5 6, obtaining the same RPN value for both the input sets is not representative enough for risk comparison between the two failures.

5. A Case Study on the Test Handler Process

In this paper, a number of RPN models (the traditional RPN, the Fuzzy RPN and its enhance models) are evaluated using a case study on a test handler process. Testing is an important step in any semiconductor manufacturing process. It plays an important role to ensure the functioning and the quality of each Integrated Circuits (IC) before being shipped out to end customers. Two commonly used equipments in the testing process include the tester which is used to test the ICs electrically, and the test handler which is used to handle the ICs while undergoing testing process. The test handler feeds the ICs to the tester through a test interface unit. Then, it classifies the tested ICs into different categories according to the tested results.

Several experiments were conducted. Experts' knowledge was collected in the FPR format initially. Refinement on the initial experts' knowledge was done by changing global weight of each rule, hence the WFPR representation.

5.1 Experiment I — Failure Risk Evaluation, Ranking, and Prioritization

Table 4. Failure risk evaluation, ranking and prioritization results using RPN, Fuzzy RPN (with both the FPR and WFRPR).

Failure	SEV	OCC	DET	RPN	RPN Rank	Fuzzy RPN	Weighted Fuzzy RPN	Fuzzy RPN rank	Weighted Fuzzy RPN rank	Expert knowledge	Expert knowledge (weighted rule base)
A	1	1	1	1	1	311	348	1	1	LOW	LOW 100%
B	1	2	1	2	2	418	416	2	2	LOW	LOW 100%
C	1	2	3	6	6	447	447	3	3	LOW	LOW 100%
D	1	3	1	3	3	463	460	4	4	LOW MEDIUM	LOW MEDIUM 100%
E	1	5	1	5	5	571	571	5	5	MEDIUM	MEDIUM 100%
F	2	2	1	4	4	607	601	6	6	HIGH MEDIUM	MEDIUM 10% HIGH MEDIUM 90%
G	1	7	1	7	7	729	717	7	7	HIGH MEDIUM	HIGH MEDIUM 100%
H	7	1	4	28	10	742	745	8	8	HIGH MEDIUM	HIGH MEDIUM 100%
I	2	9	1	18	8	868	771	9	9	HIGH	HIGH MEDIUM 60% HIGH 40%
J	2	10	1	20	9	908	787	11	10	HIGH	HIGH MEDIUM 60% HIGH 40%
K	2	10	4	80	11	905	797	10	11	HIGH	HIGH MEDIUM 50% HIGH 50%
L	6	8	9	432	12	918	837	12	12	HIGH	HIGH MEDIUM 30% HIGH 70%
M	6	9	9	486	13	918	842	12	13	HIGH	HIGH MEDIUM 20% HIGH 80%
N	7	10	10	700	14	918	869	12	14	HIGH	HIGH MEDIUM 15% HIGH 85%

This experiment evaluates and compares the traditional RPN model, the Fuzzy RPN and its improved model ability in failure risk evaluation, ranking and prioritization. Part of FMEA data from the test handler was chosen for this experiment.

Table 4 shows the results of the traditional and Fuzzy RPN (with FPR and WFPR) models for failure risk evaluation, ranking and prioritization. Columns *"SEV"*, *"OCC"* and *"DET"* (Severity, Occurrence, and Detect, respectively) are the inputs to the RPN models. Failure risk evaluation, ranking and prioritization results based on the traditional RPN model are shown in columns *"RPN"* and "*RPN rank*".

Columns *"FRPN"* and *"Weighted FRPN"* show the failure risk evaluation results using the Fuzzy RPN model with FPR and WFPR, respectively. Referring to columns *"Fuzzy RPN rank*"and *"Weighted Fuzzy RPN rank*", the results show that failures from the Fuzzy RPN model are prioritized in accordance with experts' knowledge, as shown in columns *"Expert knowledge"* and *"Expert Knowledge (Weighted Rule base)"*.

Referring to Failures C, D, E, and F (with input sets of 1 2 3, 1 3 1, 1 5 1, and 2 2 1 respectively), the traditional RPN ranks Failure D to have the lowest failure risk, following with Failures F, E, and C (with traditional RPN values of 3, 4, 5, and 6 respectively). However, failures risk ranking results for the Fuzzy RPN model in ascending order are C, D, E and F (with experts' knowledge of *Low*, *Low Medium*, *Medium*, and *High Medium;* and the associated Fuzzy RPN values of 447, 463, 571, and 607, respectively). The WFPR approach yielded the same failure risk ranking results (with expert knowledge *Low 100%*, *Low Medium 100%*, *Medium 100%* and *Medium 10% High Medium 90%*) but with different weighted Fuzzy RPN values; 447, 460, 571 and 601, respectively.

5.2 Experiment II — Study of the monotone property

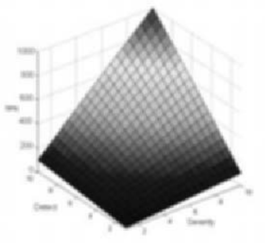


Fig. 5. The RPN versus inputs at Occurrence =10

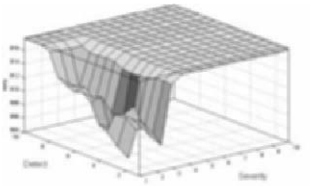


Fig. 6. The Fuzzy RPN output versus inputs at Occurrence =10.

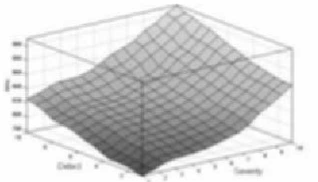


Fig. 7. The Fuzzy RPN (WFPR) versus inputs at Occurrence =10.

In this experiment, the ability of the RPN models to give a logical risk evaluation was investigated. Figures 5 to 9 show the inputs-output relationship of the RPN and Fuzzy RPN models. Figures 5 to 7 plot the RPN values versus Severity and Detect at Occurrence equals to 10.

The traditional RPN model utilizes Equation (1) and it is independent from experts' knowledge. This guarantees the traditional RPN to give a monotone output all the time as shown in Figure 5.

As compared with the traditional RPN model, the fuzzy approach does not exhibit this property. Using the Fuzzy RPN model with FPR, Figure 6 shows a non-monotone relationship between output and inputs. However, the model can be improved using WFPR, as shown in Figure 7 where a monotone output is obtained.

The monotone property can be revealed from the entries in Table 4. Referring to Table 4, Failures J and K, respectively, have input scores of 2 10 1 and 2 10 4 for Severity, Occurrence, and Detect. It is clear that Failure K should have a higher risk than Failure J. However, the Fuzzy RPN model with FPR evaluated Failure J to have a higher risk than Failure K (with the Fuzzy FRN values equal to 908 and 905, respectively). This result can be improved after the weights of the FPR were refined using WFPR (with the weighted FRPN values equal to 787 and 797, respectively).

5.3 Experiment III — Study of the output resolution property

In this experiment, the output resolution property of the Fuzzy RPN models was examined. Figures 8 and 9 show the RPN versus Occurrence and Detect at Severity equals to 1 using the Fuzzy RPN models with FPR and WFPR, respectively. The RPN values versus Severity and Occurrence at Detect equals to 4 using the fuzzy approach before and after the rules refinement phase are shown in Figures 10 and 11.

From Figures 8 and 10, we can see that FPR yielded more or less constant RPN values when Occurrence or Detect increased at the edges. From Figures 9 and 11, WFPR gave an incremental RPN values as Occurrence or Detect increased in the entire input space. This demonstrates that the WFPR representation can produce a higher output resolution of the RPN values with respect to increase in the input factors.

The output resolution property can also be observed from Table 4. With reference to Failures L, M, and N (respectively, 6 8 9, 6 9 9 and 7 10 10 (Severity, Occurrence, and Detect)), FPR produced the same RPN value (918). WFPR, however, yielded different RPN values, i.e., 837, 842, and 869, respectively, for failures L, M and N.

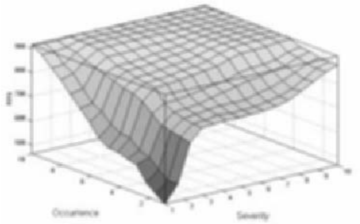


Fig. 8. The Fuzzy RPN output versus inputs at Severity =1

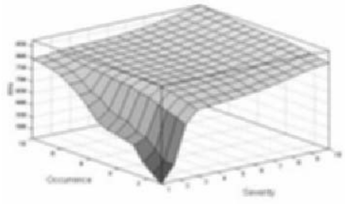


Fig. 9. The Fuzzy RPN (WFPR) versus inputs at Severity =1

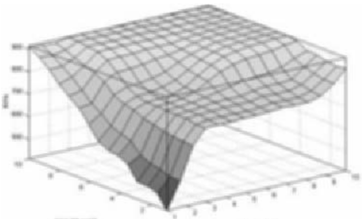


Fig. 10. The Fuzzy RPN output versus inputs at Detect =4

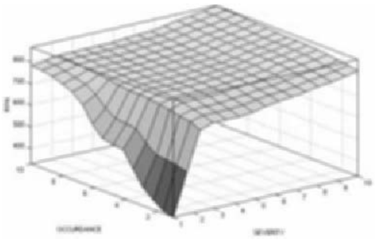


Fig. 11. The Fuzzy RPN (WFPR) versus inputs at Detect=4

6. Conclusion

In this paper, a generic approach for modeling of the fuzzy RPN function is introduced in an attempt to overcome the shortcomings of the RPN function in traditional FMEA. Comparing with the traditional methodology, the advantages of the fuzzy approach are two-fold: first it allows failure risk evaluation, ranking, and prioritization to be conducted based on experts' knowledge, experiences and opinions; second, it allows the failure risk evaluation function to be customized based on the nature of a process or a product. The proposed fuzzy approach has also been enhanced. By refining the weights associate with the FPRs, the ability of the fuzzy approach in failure risk evaluation, ranking, and prioritization can greatly be improved.

Performance of the Fuzzy RPN model has been evaluated with a real case study relating to a semiconductor manufacturing process. The experimental results positively demonstrate that the enhanced fuzzy approach allows failure risk to be evaluated logically (monotone output property) and to fulfill the output resolution property.

Further work will focus on conducting more case studies to ascertain the effectiveness of the proposed fuzzy approach with WFPR in a variety of FMEA applications. In addition, investigations on the application of learning algorithms such as neuro-learning and evolutionary computation to the Fuzzy RPN model will be carried out to realize a robust and efficient Fuzzy RPN model.

7. References

Ben-Daya, M., and Raouf, A. (1993). "A revi sed failure mode and effects analysis model," *International Journal of Quality & Reliability Management, 3(1):*43–7.

Bowles, John B. and Peláez, C. Enrique (1995), "Fuzzy logic prioritization of failures in a system failure mode, effects and criticality analysis," *Reliability Engineering & System Safety,* Vol. 50, Issue 2, Pages 203-213

Chrysler Corporation, Ford Motor Company, and General Motors Corporation (1995), *Potential Failure Mode And Effect analysis (FMEA) Reference Manual.*

Guimarães, Antonio C. F., and Lapa, CelsoMarcelo Franklin (2004), "Effects analysis fuzzy inference system in nuclear problems using approximate reasoning," *Annals of nuclear Energy,* vol 31, pp107-115.

Ireson, G., Coombs, W., Clyde, F., and Richard Y. Moss (1995). *Handbook of Reliability Engineering and Management.* McGraw-Hill Professional; 2nd edition

Jang, J. S. R., Sun, C. T., and Mizutani, E. (1997). *Neural-Fuzzy and soft Computing,* Prentice-Hall 1997.

Lin, C .T ., and Lee, C .S .G . (1995), *Neural Fuzzy Systems, A Neuro-Fuzzy Synergism to Intelligent systems.* Prentice-Hall.

Peláez, C. Enrique and Bowles, John B. (1996), "Using fuzzy cognitive maps as a system model for failure modes and effects analysis," *Information Sciences,* Volume 88, Issues 1-4, Pages 177-199.

Pillay, Anand and Wang, Jin (2003), "Modified failure mode and effects analysis using approximate reasoning," *Reliability Engineering & System Safety,* Volume 79, Issue 1, Pages 69-85.

Xu, L., Tang, L . C., Xie, M., Ho, L. H., and Zhu, M. L (2002). 'Fuzzy assessment of FMEA for engine systems," *Reliability Engineering & System Safety,* Volume 75, Issue 1, 2002, Pages 17-19.

Yeung, D. S., and Tsang, E. C. C. (1997), "Weighted fuzzy Production rules," *Fuzzy sets and Systems, v*ol.8, pp.299-313.

Bayesian Networks Approach for a Fault Detection and Isolation Case Study

G. Nunnari, F. Cannavò
Dipartimento di Ingegneria Elettrica, Elettronica e dei Sistemi
Università di Catania

Viale A. Doria, 6, 95125 Catania
Italy
[gnunnari, fcannavo]@diees.unict.it,

R. Vrânceanu
Department of Signal Circuits and Systems
The "Gh. Asachi" Technical University of Iasi
Bvd. Carol 11, Iasi 700050
Romania
radu@altastiinta.ro

Abstract. This paper presents a Fault Detection and Isolation (FDI) approach based on the use of Hybrid Dynamic Bayesian Networks (HDBN). The peculiarity of the proposed approach is that an analytical dynamic model of the process to be monitored is not required. Instead it is hypothesized that input/output measures performed on the considered process during different working conditions, including faults, are available. In the paper the proposed FDI approach is described and the performances are evaluated on synthetic and real data supplied by a standard benchmark consisting of an hydraulic actuators available in literature. The goodness of the proposed approach is assessed by using appropriate performance indices. An intercomparison between the BN approach and an other approach, namely a Multilayer Perceptron (MLP) neural network is given. Results show that the BN approach outperforms the MLP approach in some indices but it requires a high design and computational effort.

Keywords: Fault Detection and Isolation, Dynamic Bayesian networks, DAMADICS.

1 Introduction

Modern industrial processes involve several complex hardware elements. Faults or malfunctions of these elements, which unfortunately appear relatively often, can cause long-term process disturbs or even sometimes forces the installation to shut down. Moreover, the elements faults may influence the final control quality, being the source of potential reasonable economic loses.

Because in general the actuators cannot be installed but in harsh environments (high temperature, pressure, humidity, pollution, chemical solvents, aggressive media, vibration, etc.), fault detection or prediction algorithms are necessary to be applied. Continuously or periodically performed diagnosis of actuators can also cut the maintenance cost.

Although the problem of Fault Detection and Isolation (FDI) has vital importance for industrial applications it remains a complex task. Several authors have proposed a large variety of techniques, based upon both physical and analytical redundancy (Simani et al. 2002), which can be roughly classified as model based and non-model based. Despite model based approaches are widely studied they cannot be applied when an accurate analytical model of the plant is not available. For this reason other approaches have been studied, mainly investigating in the Artificial Intelligence field.

We know that uncertainty is all around us and probabilistic theory is the method of choice for dealing with uncertainty in many science and engineering disciplines such as control, physics and economics. It provides us a simple and clean semantics of how to maintain our beliefs in face of partial information and how to update them when presented with new evidence.

In this perspective several fault-symptom relations are not deterministic but rather probabilistic. Indeed, given a fault, the symptom may be detected with a certain probability, and, moreover, probability may change depending on particular working conditions. It is straightforward to conclude that probability theory provides an interesting approach to solve FDI problems. Among probability theory based approaches Bayesian networks (BN) represent a powerful tool to dealing at the same time with uncertainty, probability and graph theory (Karny et al. 1999). In particular Bayesian Networks has been usefully considered as a very attractive alternative technology for diagnostic decision tools by (Kang et al. 1999) for complex nuclear power systems and by (Lerner et al. 2000) for chemical plants. Others applications are described in (Przytula et al. 2000) and (Karny et al. 1999). To further investigate the peculiarities of the BNs based approach, in this paper we report some results concerning the application of Dynamic Bayesian Networks (DBN) to a benchmark concerning hydraulic actuators installed on particular industrial process. The performances of the proposed BN approach are evaluated according to a predefined set of indices are compared with that obtained by using a Multi-layer Perceptron based approach (MLP). In particular in this paper we show the results obtained by using the BN-FDI technique already proposed in (Nunnari et al. 2004) both on synthetic and real data of the considered benchmark.

2 Bayesian Networks Approach for FDI

The purpose of this section is to clarify the basic terms and to describe how to formulate a FDI problem in terms of Bayesian networks.

A Bayesian network, also referred as a belief network or a causal network in literature, is a graphical representation of relationships within a problem domain (Przytula et al. 2000). It is a powerful tool to deal with uncertainty representing a probability distribution by using a graphical model of a directed, acyclic graph (DAG). Every node in the graph corresponds to a random variable in the domain and is annotated by a Conditional Probability Distribution (CPD), defining the conditional distribution of the variable given its parents. The peculiarity of a Bayesian network is to compute joint probability distributions over a set of random variables. The conditional independences shown in DAG simplifies the computation of joint probability distribution. Diagnostic inferences are also possible by Bayesian Networks getting conditional probability when some variable values are known (priors). In a BN for FDI, priors are typically represented by measure-system, in this approach, is described by stochastic time series of I/O measures. Its behavior is described in terms of a system state which evolves stochastically at discrete time steps. The system state is modeled by a set of random variables X. Physical systems commonly comprise both continuous and discrete quantities. To introduce the time variable in the framework of probabilistic models, Dynamic Hybrid Bayesian Networks (HDBN) are needed, they consist of a BN and a transition influence graph which show the influences among the variables in two consecutive time slices. Each measured variable and other additional knowledge (e.g. due to human experts, other FDI models etc) are represented by random nodes. If a given variable is continuous the node is also continuous and the probability distribution is supposed to be Gaussian. If the variable is discrete the node is also a discrete ones and the probability is supposed uniformly distributed, unless other information is available. Variables that represent failure nodes of the system are typically Boolean, the two states representing the normal and fault behavior respectively. To build a dynamic net it is necessary to insert a certain number of regression nodes representing the values of given variables at previous time instants. The number of regressions for each variable depends on the particular dynamic system considered. A crucial step is the FDI process is to find a DAG. The influence arcs are inserted on the base of a-priori information, heuristic knowledge, temporal causal graph of the system, or evidences obtained from the structure learning algorithms. In a simplistic network the fault node influences all measures.

The hybrid dynamic Bayesian network (HDBN) represents the complete fault state of the system at each time step (Neapolitan 2004). A fault variable in belief state takes into consideration all the evidences available up to the present time in order to determine a probability distribution of a residual for the considered fault.

3 Benchmark Specifications

In order to test the proposed algorithm a suitable benchmark has been considered. The benchmark is known in the literature as DAMADICS (Development and Application of Methods for Actuator Diagnosis in Industrial Control Systems); it is a public domain benchmark appropriately studied to provide the possibility of evaluating and ranking research approaches. The benchmark was developed in order to compare the properties of Fault Detection and Isolation (FDI) methods intended to be applied for actuator diagnosis in industrial environment; it concentrates in particularly on a kind of hydraulic actuator installed on a sugar plant. A well documented description of the benchmark can be seen and downloaded from DAMADICS Information Website (Ref. DAMADICS website).

The benchmark deals with actuators installed in a Lublin Sugar Factory (Poland) and its main goal is the generation of well defined, repeatable single actuator faults. For DAMADICS project two benchmarks were prepared: a model based (software) and a data based one (real industrial process data). Analytical model of actuator is not available. Due to unlimited number of possible fault scenarios the important limitations were introduced: number of considered actuator faults is equal to 19, two fault simulation scenarios are assumed: abrupt and incipient, only single fault scenarios are considered and simulated.

Actuator or *final control element* is a physical device, structure or assembly of devices acting on controlled process. Taking into account a benchmark definition it will further understand the actuator as a set consisting of: control valve, spring-and-diaphragm pneumatic servomotor and positioner

Refering to the benchmark the following variables are considered: hand driven bypass valve (V1), hand driven bypass valve (V2), hand driven bypass valve (V3), control valve (V), pressure sensor (valve inlet P1), pressure sensor (valve outlet, P2), process media flowmeter (F), valve displacement (X), positioner feedback signal (x), pneumatic servo-motor supply (ps), control signal from external PI controller (CV). Three actuators have been chosen for research purposes in the framework of DAMADICS project [Was02].

In addition to previous limitations in this paper we have considered the following restriction: results refer to abrupt faults only. Moreover, for the lack of pages we reports results concerning two different kinds of trials, one regarding the test over syntethic data, and the other concerning the test on real data.

In the first part of trials six of the ninentheen possible faults have been considered. According with the symbols considered in the original benchmark, the six considered faults will be referred as: f1 – valve clogging, f7 – medium cavity or critical flow, f15 – positioner supply pressure drop, f17– pressure difference drop on valve, f18 – fully or partly opened bypass valve, f19 – flow rate sensor fault.

In the second part of trials, only the four fault truly generated have been considered: f16 – positioner supply pressure drop, f17– pressure difference drop on valve, f18 – fully or partly opened bypass valve, f19 – flow rate sensor fault

4 Performance Indices

In order to evaluate the performance of the considered FDI approaches, an appropriate set of indices were considered whose definition is reported below according with the symbols in Figure 1

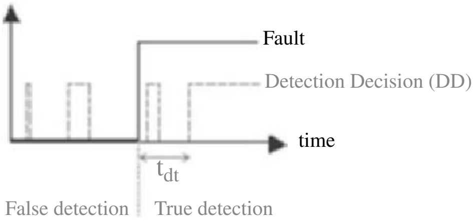


Figure 1. Explanation of parameters used in performance indices

Detection time (t_{dt}): period of time from the begin of fault start-up to the moment of the last leading edge of the detection decision (DD) signal.

False detection rate (r_{fd}): $r_{fd} = \frac{\sum_i t_{fd}^{i,DD}}{t_{from} - t_{on}}$

where $t_{fd}^{i,DD}$ is a i-th period of high DD signal value between t_{on} to t_{from}.

True detection rate (r_{td}): $r_{td} = \frac{\sum_i t_{td}^{i,DD}}{t_{hor} - t_{from}}$

where $t_{td}^{i,DD}$ is a i-th period of high DD signal value between t_{from} to $t_{hor.}$

Isolation time (t_{it}): Period of time from the begin of fault start-up to the moment of the last leading edge of the correct isolation signal (Primary Isolation Decision, PID).

False isolation rate (r_{fi}): $r_{fi} = \frac{\sum_i t_{fi}^{i,PID}}{t_{from} - t_{on}}$

where $t_{fi}^{i,DD}$ is a i-th period of high PID signal value between t_{on} to t_{from}.

True isolation rate (r_{ti}): $r_{ti} = \frac{\sum_i t_{ti}^{i,PID}}{t_{hor} - t_{from}}$

It is straightforward to understand that the best situations in terms of t_{dt} (or t_{it}) are obtained when its value approach zero. The same consideration can be made for r_{fd} (or r_{fi}), while the best value for r_{td} (or r_{ti}) is 1.

5 The FDI System

The FDI problem that is proposes consists in detecting and isolating no less than 19 faults that are likely to occur in an actuator. Because a non-model based approach will be considered, the four inputs (CV, P1, P2, T) and the two outputs (F,X) are the only information that we have about the behaviour of the system, with or without a certain fault. Some several BN were designed and built to get as close as possible to this behaviour, the most significant ones will be presented in the rest of the paper. As in any non-model based method a certain amount of data is necessary for both training and testing; the data can be either synthetic (generated) or real (usually provided). For a better evaluation of the performances in this work both types of data will be considered and the results will be described and commented. Also a comparison with other approaches (basically neural-networks) will be reported.

5.1 Train and Test Synthetic Data

For generating synthetic data for both training and testing the Damadics Actuator Benchmark Library has been used. It consists from several blocks made in Matlab-Simulink designed to act like the real actuator and to provide a background for creating the data. All the synthetic data is automatically generated in the range of 0 to 1 while for the real data the maximum values are provides. For testing the

DGEN block has been used to simulate a normal behaviour of the actuator for 900 seconds, after which, depending on its settings, it reproduces the behaviour altered by one of the nineteen faults with certain intensity. We considered being sufficient 2000 seconds for testing (900+1100) so, all the data used has this size (this is possible due to the fact that only abrupt faults were considered). As mentioned before, because the actuators work in various conditions with different working points, the data used both training and testing patterns had to be generated accordingly. To satisfy this requirement, in the scheme above, the frequencies of P1, P2 and CV have been chosen as prime numbers. In addition CV was varied from 0.3 to 0.7 through sinusoidal function with frequency of 0.01 Hz. The same function was also used to vary P1 from 0.83 to 0.92 with frequency of 0.51 Hz and P2 from 0.64 to 0.66 with frequency of 0.57 Hz. Furthermore, the phase of the sinusoidal functions have been changed from fault to fault pattern. Data for all abrupt faults was generated in this way. A simulation time of 1500 seconds was considered to be sufficient for the training process. In both training and testing process, for all BNs only 4 regressions were considered and the regression time slice was chosen to be 15 seconds.

5.2 FDI on Synthetic Data

Because of the nature of the variables involved in the DAMADICS benchmark, some simplifications can be made. They consist in combining 2 pairs of variables in just two nodes (instead of four). That is possible because, on one hand, CV tries to follow X and so only their difference is significant, and, on the other, fluctuations of both P1 and P2 regard only one fault: F17, in all the other cases them being constant (so they too can be combined). To build the Bayesian net all fault have been introduced and they have been linked only to nodes that the relative fault influences. This approach allows the extent of the network for relatively many faults. To prove this a network was designed to deal with the six-faults scenario that was proposed before. It's structure is presented in Figure 2a: As a novel technique the *EA* node is inserted to bring out the powerful "explaining-away" phenomenon. The simulation was carried out in two steps: first training and then testing. In order to evaluate the goodness of our approach, it was compared with two other non-model based FDI techniques, in particular with a Neural approach. The MLP neural networks used for comparison were trained as classifiers. In particular six classifiers were implemented, each devoted to detect and isolate one of the six different fault classes. MLP with one hidden layer were considered and the best number of neurons in the hidden layer was determined by a trial-and-error approach; this number was chosen to be 6 and adopted for all the considered classifiers. The number of input neurons was set to 18 since the number of input variable is 6 (i.e. CV, P1, P2, X, F, T) and for each of them 3 regression were considered. The regression time slice was chosen to be 20 seconds. Bearing in mind the above considerations it is easy to conclude that the structure of each classifier was set to 18-6-1 (i.e. 18 neurons in the input layer, 6 in the hidden layer and 1 in the output layer). In order to train a MLP classifier to detect and isolate a given fault class, the training patterns were obtained setting the output values to 0 if the pattern represents the fault-free class or if it belongs to a different fault class, otherwise it was set to 1 (just like for BNs). In this way, each MLP can detect only one fault class. The results are summarized in Table 1. They are to compare to the ones obtained by the NN (Table 2) :

Table 1. Best performances indexes obtained by the BN approach

Fault	$\mathbf{t_{dt}}$**[s]**	$\mathbf{r_{fd}}$	$\mathbf{r_{td}}$	$\mathbf{t_{it}}$**[s]**	$\mathbf{r_{fi}}$	$\mathbf{r_{ti}}$
F1	0	0.057	1	0	0.057	1
F7	0	0	1	0	0	1
F15	4	0.004	0.996	4	0.004	0.996
F17	0	0	1	0	0	1
F18	0	0.078	1	0	0.078	1
F19	ND	0	0	ND	0	0

Table 2. Best performance indexes obtained by the neural classifier.

Fault	$\mathbf{t_{dt}}$**[s]**	$\mathbf{r_{fd}}$	$\mathbf{r_{td}}$	$\mathbf{t_{it}}$**[s]**	$\mathbf{r_{fi}}$	$\mathbf{r_{ti}}$
F1	23	0	0.979	23	0	0.979
F7	75	0	0.981	75	0	0.981
F15	14	0	0.97	14	0	0.97
F17	0	0.002	1	0	0.002	1
F18	11	0	0.859	11	0	0.859
F19	ND	0	0	ND	0	0

As a general comment it is possible to say that all the two inter-compared techniques performed quite well. However a ranking of these techniques in terms of t_{dt}, r_{td}, t_{it} and r_{ti} points out that the BN approach outperform the MLP while this latter techniques works better than BN in terms of r_{td} and r_{fi}.

5.3 FDI on Real Data

This step proposed is based on real data from the three actuators installed in the Lubin Sugar. To generate the real data, only external faults had been considered (due to the difficulty of dealing with other real faults); this means that only faults F16, F17, F18 and F19 have been paid attention. That is why, the networks that were designed and used, dealt only with these four faults. In order to evaluate again the performances of the BNs, a new comparison was made with a neural network. The networks had to be redesigned to deal only with these four faults. Their performances will be presented in Table 3 and discussed a little later. The Bayesian network used for testing on real data has the structure as shown in Figure 2b.

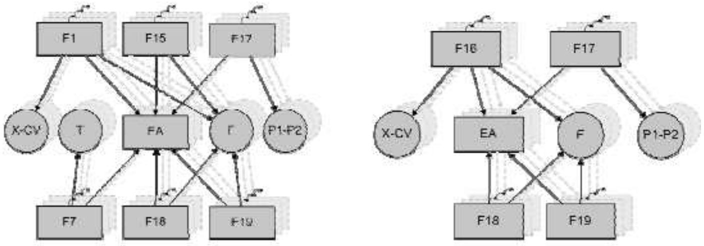


Figure 2.(a) An improved network used for the 6 faults considered in the comparison with the neural networks (b) The BN used for testing on real data.

As it can be easily seen, it follows the same principles discussed earlier for the other network: the faults are modelled as discrete variables with only two states (true and false) with a tabular type distribution while the measurements are represented as continuous nodes with a Gaussian CPD. It is also to be noticed the "explaining away" (EA) node whose role has been described earlier. Once again, only the nodes which model the four faults are correlated in time (as shown in the figure above with curved arrows). The network was trained with the synthetic data. The network performed pretty well in terms of detection time and isolability while not above average in terms of number of detected faults. As it will be later described in Table 3, 8 out of 19 faults were successfully detected (all six faults F16, one F17 and one F18) with detection times ranging from 3 to 38 seconds, and 100% isolability. An example of such detection is shown in the next figure:

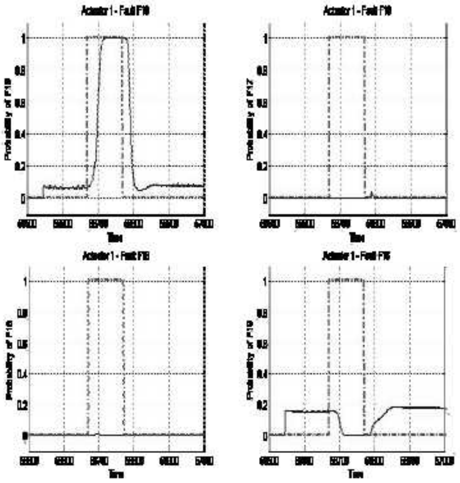


Figure 3. Results for testing on real data; here for fault nr. 16

The presented result in Figure 3 is neither the best nor the worst that the network had, it is just an average one. The fault was simulated for Actuator 1 on the date of November 17, 2001 and it was a positioner supply pressure drop (F16).

The BN correctly detected and isolated it in no more than 30 seconds. The probability of being fault F16 raised up to one while the others remained to zero. This case was chosen not for its outstanding performances but for its very nice example of the "explaining away" phenomenon; even if F19 presents a low probability (about 0.18) of existence, as soon as the network realizes that the fault is actually F16, the probability of F19 becomes practically zero. Once again, this is due to the fact that the presence of F16 explains the observed data and the behaviour of the system.

The neural network designed to deal with the real data was considered to be Artificial neural networks can be successfully used as a different non-model based technique for FDI. In particular in this case a Multilayer Perceptron has been implemented in order to classify the fault patterns. The network structure was chosen after a trial-and-error process. The input variables were three (CV-X, P1-P2, F) and the number of regressors was fixed to five, so the final network has 15 input neurons. The network has an hidden layer with 6 neurons, and 4 neurons in the output layer. Each output node is devote to indicate the fault occurrences of a single class. The patterns used in learning phase contain all four faults simulated by DGEN block. In each pattern no more than one fault is present. In testing phase the threshold used to discriminate the fault patterns was set at 0.8

After some trials we have noticed that the insertion, in learning, of patterns regarding F19 degrades results. During the test the network has exhibited no ambiguous signals, in other word in presence of a fault all outputs except for one were under threshold. This means that all detected faults have been isolated. The results for all three approaches are summarized in the table below:

Table 3. Comparison between two BN and one NN approaches. (- not detectable)

Fault	**Actuator**	**Date**	**Detection Time [s]**		**Isolability**		**True Detect. Rate**	
			BN	**NN**	**BN**	**NN**	**BN**	**NN**
F18	1	30/10/01	-	-	-	-	-	-
F16	1	9/11/01	35	31	ok	ok	0.87	0.865
F18	1	9/11/01	-	-	-	-	-	-
F18	1	9/11/01	-	42	-	ok	-	0.247
F18	1	17/11/01	-	32	-	ok	-	0.673
F16	1	17/11/01	30	28	ok	ok	0.70	0.712
F17	1	20/11/01	3	13	ok	ok	0.99	0.99
F17	2	17/11/01	-	-	-	-	-	-
F17	2	17/11/01	-	-	-	-	-	-
F19	2	17/11/01	-	-	-	-	-	-
F19	2	17/11/01	-	-	-	-	-	-
F19	2	17/11/01	-	-	-	-	-	-
F17	2	20/11/01	-	-	-	-	-	-
F18	3	30/10/01	38	-	ok	-	0.32	-
F16	3	9/11/01	15	14	ok	ok	0.70	0.732
F16	3	9/11/01	16	20	ok	ok	0.82	0.732
F16	3	9/11/01	18	24	ok	ok	0.67	0.708
F16	3	17/11/01	17	28	ok	ok	0.86	0.708
F19	3	17/11/01	-	-	-	-	-	-

In the Table 3 we have used the following notations: BN is the Bayesian Network designed to deal with real data while NN refers to the neural network. It can be easily noticed that BN and NN have almost the same performances concerning both isolability and detection time. However, the NN outperformed the BN in terms of number of faults detected; while NN has discovered 10 out of 19, BN_1 couldn't find more than 8 of them. On the other side, the isolability of the faults was clearer for BN (e.g. for NN a 0.8 value had to be used as threshold for detection, a smaller value leading to unisolability). It is also to be noticed that the neural network had to be trained without F19, otherwise giving much worse results. As another comment, the false isolation rate was more or less 0.1% for BN while for NN is raised up to 3%. The bad performances for F19 (neither BN and NN were able to detect it) can be explained considering that it is a sensor fault (to be more precise a flow rate sensor fault) and that the networks couldn't learn from the train data the pattern of the fault.

6 Conclusions

In this paper a Bayesian Network FDI based approach has been proposed and applied to a benchmark considered in literature consisting of a hydraulic actuators which is a subsystem of several industrial plants. The results shows that the implemented systems performs quite well in terms of the most common performance indices usually considered for such a kind of application. The comparison with other techniques traditionally considered for FDI applications shows that there could be some advantage in terms of performances but there are also some drawbacks in terms of design and computational effort. However it is to be stressed that the results reported in this paper have been obtained working on data generated by a SIMULINK model of the real system. This means the further confirmation about the reliability of the proposed approach on real process data. Work is in progress to this end.

Acknowledgments

This paper has been supported by the research project "Rivelazione e diagnosi di malfunzionamenti, riconfigurazione del controllo e monitoraggio delle prestazioni nei processi industriali, PRIN (Progetti di Rilevante Interesse Nazionale) 2002-2003, Ministero dell'Istruzione, dell'Università e della Ricerca.

References

DAMADICS website http://diag.mchtr.pw.edu.pl/damadics .

Kang C.W. ; Golay M.W., *A Bayesian belief network-based advisory system for operational availability focused diagnosis of complex nuclear power systems*, Expert Systems with Applications , volume 17, pages 21--32, 1999.

Karny, M., Hangos K., Tesar L.; *On probabilistic modeling in fault detection*, European Control Conference. ECC '99, Karlsruhe, 1999.

Lerner U., Parr R., Koller D., Biswas B., *Bayesian fault detection and diagnosis in dynamic*, in AAAI/IAAI, pages 531--537, 2000.

Neapolitan R. E., *Learning Bayesian Neyworks,* Prentice-Hall, 2004, pp. 205-265

Nunnari G., Cannavo' F., Vranceanu R., *Fault Detection and Isolation by Using Bayesian Networks: a Case of Study*, Proc. ISDA pages 861-866, 2004

Przytula, K.W.; Thompson, D., *Construction of Bayesian networks for diagnostics*, Aerospace Conference Proceedings, 2000 IEEE , volume 5, pages 193--200, March 2000.

Simani, S., Fantuzzi C., Patton R.J., *Model-base Fault Diagnosis in Dynamic Systems Using Identification Techniques*, Springer, 2002.

Part V

Tracking and Surveillance

Path Planning Optimization for Mobile Robots Based on Bacteria Colony Approach

Cezar Augusto Sierakowski and Leandro dos Santos Coelho

Pontifícia Universidade Católica do Paraná, Grupo Produtrônica
Programa de Pós-Graduação em Engenharia de Produção e Sistemas
Rua Imaculada Conceição, 1155, CEP 80215-901, Curitiba, PR – Brazil
E-mail: cezars@globo.com; leandro.coelho@pucpr.br

Abstract. Foraging theory originated in attempts to address puzzling findings that arose in ethological studies of food seeking and prey selection among animals. The potential utilization of biomimicry of social foraging strategies to develop advanced controllers and cooperative control strategies for autonomous vehicles is an emergent research topic. The activity of foraging can be focused as an optimization process. In this paper, a bacterial foraging approach for path planning of mobile robots is presented. Two cases study of static environment with obstacles are presented and evaluated. Simulation results show the performance of the bacterial foraging in different environments in the planned trajectories.

1 Introduction

Swarm intelligence is an emerging research area with similar population and evolution characteristics to those of genetic algorithms. However, it differentiates in emphasizing the cooperative behavior among group members. Swarm intelligence is used to solve optimization and cooperative problems among intelligent agents, mainly in computer's networks, mobile robotics [11] and cooperative and/or decentralized control [2], [7], [10], [16]. Swarm intelligence, in nature, may be composed of three main principles: evaluation, comparing and imitation. Evaluation is the capacity to analyze what is positive or negative in nature, attractive or repulsive. Even the smaller life forms have these abilities, in the case of bacteria, they are able to notice if the environment in which they are located is noxious or not. Learning won't happen unless beings are capable of evaluate the attractive and repulsive characteristics of the environment. Comparison is the way living beings use other beings as a standard to evaluate themselves, results of these comparisons may become a motivation to learning and/or modification. Imitation is an effective form of learning. However, very few animals, in nature, are capable of imitating, in fact, only human beings and some species of birds are capable of such action [10]. These three basic principles may be combined, in a simplified version, in computer programs, opening possibilities for them to adapt to complex problems. Animals, or groups of animals, when foraging, act looking for maximizing the amount of energy obtained per unit of time spent foraging, considering the biological and environmental limitations.

In the context of biologically inspired optimization methods, several models of bacterial chemotaxis algorithm based on pioneered work of Bremermann [4] have been proposed in literature for applications in biology [1],[13], genetics [5], communication networks [20] and robotics [6]. In this paper, the foraging theory is applied to bacteria, adopting the bacteria colonies nomenclature. The fact that bacteria are one of the simplest living beings existing in Earth and they use the forage theory to benefit the group motivated this study. Bacteria own a control system that allows the foraging control and avoidance of noxious substances. In this context, the cooperation activity in a bacteria colony may be used in an optimization procedure, based in the forage strategy, as proposed by Passino [15].

This paper's contribution is to present a bacteria colony study for planning mobile robots trajectory. In this context are evaluated simulated cases of an environment composed by several obstacles. The next sections of the paper are organized as follows. Section 2, fundamentals and equations for bacteria colonies algorithm are discussed. Section 3 presents analysis of mobile robots trajectory optimization problem and cases. In Section 4 is discussed the results obtained with the bacteria colonies application. Conclusions and future perspectives are shown in Section 5.

2 Bacteria Colony

Natural Selection tends to eliminate animals with poor foraging strategies and to favor gene propagation of those with good foraging strategies, once these have higher chances of succeeding in reproduction. These evolutionary principles have taken scientists to develop the foraging strategies, turning it appropriate to optimization models [15]. The presence of flagellum allows the bacteria to move, the movement is acquired through the flagellum rotation in the same direction, at a rotating speed of 100 to 200 rotations per second. Bacteria may move in two different forms: they might run (swim for a period of time), movement achieved by the flagellum rotation counterclockwise, or they can tumble, achieved by the flagellum rotation clockwise. Bacteria switch between these two modes of operation during its entire lifetime (rarely the flagellum stops rotating).

After a run period, a bacterium tumbles, the tumble interval is about $0{,}14 \pm 0{,}19s$, according to Passino [15]. After the tumble, the bacterium is pointed in a random direction. When the flagellum are rotated counterclockwise, the bacterium will move towards the direction it's turned, at an average speed of $10-20\mu m/s$, meaning, about 10 times its length by second, for a mean interval of $0{,}86 \pm 1{,}18s$. The local environment where bacteria live might change, either gradually or suddenly. So bacteria can suffer a process of elimination, through the appearance of a noxious substance, or to disperse, through the action of another substance, generating the effects of elimination and dispersion.

A bacterium position, in a time instant, can be determined through equation (1), where the position in that instant is calculated in terms of the position in the previous instant and the step size $C(i)$ applied in a random direction $\phi(j)$, generated in the bacterium tumble,

$$\theta'(j+1,k,l)=\theta'(j,k,l)+C(i)*\phi(j,k,l) \tag{1}$$

To adapt such strategy to optimization problems, an equation to determinate the cost of each position is needed to enable the comparison between the position and the environment. The cost is determined by the equation,

$$J(i,j,k,l)=J(i,j,k,l)+J_{cc}(\theta'(j,k,l),P(j,k,l)) \tag{2}$$

Through equation (2) is noticed that the cost of a determined position $J(i,j,k,l)$ is also affected by the attractive and repulsive forces existing among the diverse bacteria of the population $J_{cc}(\theta'(j,k,l),P(j,k,l))$.

After a determined number of chemotactic steps (steps comprehending the movement and the cost determination of each bacterium position), a reproductive step occurs. In this reproductive step the bacterium are sorted decreasingly by their cumulative cost. The lower half of the list die, these are the bacteria that couldn't gather enough nutrients during the chemotactic steps, and the upper half divide themselves into two new bacteria, located in the same position.

Summarizing, the term *taxis* refers to the locomotory response of a cell to its environment. In a taxis, a cell responds such that it changes both direction and the duration of the next movement step. The tactic response requires some directional information form the environment that bacteria obtain by comparing an environmental property at two different time steps. If the tactic response is related to information about chemical concentrations (that may be either attractants or repellents), it is called chemotaxis [14].

In Figure 1 (shown at the end of the article) the algorithm is presented in pseudo code. As seen in the pseudo code, the bacteria colony algorithm is basically composed by an elimination and dispersal loop, inside this loop, there is another one, who is responsible for the bacteria reproduction. Inside this one, there is a third loop, responsible for generating the direction in which each bacterium will run, determining the period the bacterium will move and, as a consequence, determining it's position after the loop execution, and calculating the fitness of these positions. The reproductive loop is responsible for determining which of the bacteria must reproduce and which must be exterminated after the movements executed in loop 3, through a cost analysis of their positions along their movement. The first loop is responsible for eliminating some bacteria; it's ruled by an elimination probability, repositioning them into another random position of the search space.

```
 DO{ $l = l + 1$;
  DO{ $k = k + 1$;
    DO{ $j = j + 1$;
      FOR EACH bacterium $i$ {
        Calculate $J(j,k,l)$; Assume $J(j,k,l) = J(j,k,l) + J_{cc}(\phi'(j,k,l), P(j,k,l))$;
        Save $J_{last} = J(i,j,k,l)$;
        Generate a random vector $\Delta(i) \in \Re^{p}$, with real numbers, within $[-1,1]$;
        Move $\theta'(j+1,k,l) = \theta'(j,k,l) + C(i) * \frac{\Delta(i)}{\sqrt{\Delta^{\tau}(i) * \Delta(i)}}$;
       Calculate $J(i,j+1,k,l)$;

        Assume $J(i,j+1,k,l) = J(i,j+1,k,l) + J_{cc}(\theta'(j+1,k,l), P(j+1,k,l))$;
        Assume $m = 0$;
        DO{
          Assume $m = m + 1$;
          IF $J(i,j+1,k,l) > J_{last}$  (optimization problem) THEN
            Assume $J_{last} = J(i,j+1,k,l)$;
            Calculate $\theta'(j+1,k,l) = \theta'(j+1,k,l) + C(i) * \frac{\Delta(i)}{\sqrt{\Delta^{\tau}(i) * \Delta(i)}}$;
            Calculate $J(i,j+1,k,l)$;
            Assume
               $J(i,j+1,k,l) = J(i,j+1,k,l) + +J_{cc}(\theta'(j+1,k,l), P(j+1,k,l))$;
          ELSE Assume $m = N_s$;
        } WHILE $m < N_s$;
      }
    } WHILE $j < N_c$;
    FOR EACH bacterium $i$ {
      Calculate $J^{i}_{Health} = \sum_{j=1}^{N_s - 1} J(i,j,k,l)$;}

      Sort the bacteria, according to the valor of $J_{Health}$;

      Kill the bacteria with the smaller value of $J_{Health}$;

      Duplicate the bacteria with the higher values of $J_{Health}$;
  } WHILE $k < N_{re}$;
   FOR EACH bacterium $i$ {
      Eliminate and disperse bacterium with a probability of $p_{ed}$;}
} WHILE $l < N_{ed}$.
```

Fig. 1. Pseudo code of the foraging theory applied to a bacteria colony.

3 Trajectory Planning of Mobile Robots

Literature is rich in boardings to solve mobile robots trajectory planning in presence of static and/or dynamic obstacles [18], [3], [12]. One of the most popular planning methods is the artificial potential fields [17]. However, this method gives only one trajectory solution that may not be the smaller trajectory in a static environment. The main difficulties in determining the optimum trajectory are due to the fact that analytical methods are extremely complex to be used in real time, and the searching enumerative methods are excessively affected by the size of the searching space.

Recently, the interest in using evolutionary algorithms has increased, genetic algorithms are used in mobile robots trajectory planning, generally when the search space is large [8], [19], [9]. The trajectory planning is the main aspect in the movement of a mobile robot. The problem of a mobile robot trajectory planning is typically formulated as follows: given a robot and the environment description, a trajectory is planned between two specific locations which is free of collisions and is satisfactory in a certain performance criteria [19].

Seeing the trajectory planning as an optimization problem is the boarding adopted in this article. In this case, a sequence of configurations that moves the robot from an initial position (origin) to a final position (target) is designed.

A trajectory optimizer must locate a series of configurations that avoid collisions among the robot(s) and the obstacle(s) existing in the environment. The optimizer must also try to minimize the trajectory length found, in order to be efficient. The search space is the group of all possible configurations.

In the present study, it's considered a bidimensional mobile robot trajectory planning problem, in which the position of the mobile robot *R* is represented by Cartesian coordinates *(x,y)* in the *xy* plan. The initial and destination points of the robot are (x_0, y_0) and (x_{np}, y_{np}), where n_p is a project parameter. The initial point is always (100,100).

Only the trajectory planning problem is empathized in this paper, the robot control problem is not the focus ofit this paper. However, details of the robots movement equations can be found in Fujimori *et al.* [8]. It's assumed that the obstacles are circular in the robot's moving plan. Besides, the hypothesis that the free bidimensional space is connected and the obstacles are finite in size and they do not overlap the destiny point is true.

The optimization problem formulated consists of a discrete optimization problem, where the objective function *f(x,y)*, which is the connection between the bacteria colony and the environment, aims to minimize the total trajectory travel by the mobile robot is given by

$$f(x,y) = \alpha d_{obj} + \lambda n_o \tag{3}$$

$$d_{obj} = \sum_{i=0}^{n_p} \sqrt{\left(x(i+1) - x(i)\right)^2 + \left(y(i+1) - y(i)\right)^2} \tag{4}$$

where α and λ are weighted factors, d_{obj} represents the Euclidian distance between the initial and the destiny point, n_0 denotes the number of obstacles hitten by the robot movement following the planned trajectory, and n_p is the number of points where a trajectory change occurs (project parameter in this article). It's noticed by the equation (3) that an λ term exists, it's a weighted (penalty) term for unfeasible solutions, meaning, the trajectory that intercepts obstacles. In this case, the fitness function to be evaluated by the bacteria colony aims to maximize

$$fitness = \frac{K_c}{f(x,y)+\varepsilon} \tag{5}$$

where K_c and ε are scale constants.

4 Simulation Results

In order to start the system, the parameters p (optimization problem's dimension), S (population size), N_c (number of chemotactic steps), N_s (maximum number of steps that a bacterium can swim in a turn), N_{re} (number of reproductions), N_{ed} (number of elimination-dispersals events), p_{ed} (elimination-dispersal probability) and $C(i), i=1,2,...S$ (speed of the movement taken in one step). The starting values for $\theta^i, i=1,2,...S$ (positions) where chosen randomly with normal distribution. In this paper is adopted, S = 50, N_c = 15, N_s = 4, N_{re} = 4, N_{ed} = 2, p_{ed} = 0.3 and $C(i)$ = 2.5, i=1,2,...S.

The environment used for the trajectory planning is a 100x100 meters field. The search interval of the parameters using the bacteria colony is $x_i \in [0,100]$ meters and $y_i \in [0,100]$ m, where i=1,..,np. About the fitness it's adopted α=1, λ=200, K_c =100 and ε=1x10^{-6}. Afterwards are presented two simulated cases and the results analysis after 10 experiments with the bacteria colony algorithm.

As a comparative study a genetic algorithm is also applied in the simulated cases, the genetic algorithm is configured with the selected parameters, population size 50, crossover probability 0.85, mutation probability 0.15, maximum number of generations 300, the selection operator is the roulette wheeling, and the size of each chromosome is set to 16.

4.1 Case study 1: Environment with 4 obstacles

In Table 1 are presented the positions of the centers (x_c, y_c) of the circular obstacles and their respective radius (in meters) of case 1. The results obtained with the bacteria colony are restricted to p=3. In Tables 2 and 3 the achieved solutions are presented.

Table 1. Definition of obstacles for the case study 1.

Obstacle nº	Radius	Position (x_c, y_c)
1	10	(40, 15)
2	10	(20, 35)
3	20	(75, 60)
4	15	(35, 75)

Table 2. Results for an environment with 4 obstacles for the bacteria colony.

Experiment	Mean fitness	Maximum fitness	Minimum fitness	Standard deviation
1	0.2141	0.6950	0.0513	0.1044
2	0.2047	0.6902	0.0566	0.1099
3	0.2166	0.6916	0.0572	0.1036
4	0.2089	0.6906	0.0527	0.1059
5	0.2134	0.6912	0.0525	0.0986
6	0.2054	0.6934	0.0559	0.0869
7	0.2107	0.6889	0.0567	0.0910
8	**0.2243**	0.6915	**0.0577**	**0.1213**
9	0.2082	0.6908	0.0522	0.1032
10	0.2154	**0.6952**	0.0570	0.0932

The best (higher) fitness that the bacteria colony has achieved, for p=3, has been obtained with the solution: (x_1, y_1) = (64.8631; 79.6103), (x_2, y_2) = (37.2277; 45.1039) and (x_3, y_3) = (28.3488; 28.6159). In Figure 2 the best result of the experiments is presented. In case study 1, the best of bacteria colony obtains a distance total of path of 143.8403. This distance is 98.31% of optimum path without obstacles. From 10 repeated simulations, it is shown that a robust convergence of path planning is obtained after about 120 generations (see Figure 2).

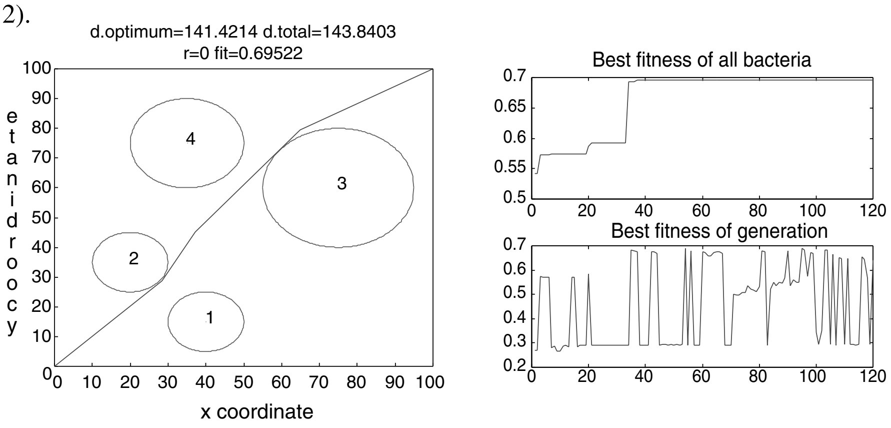


Fig. 2. Best result achieved for the bacteria colony in case study 1.

Table 3: Results for an environment with 4 obstacles for the genetic algorithm.

Experiment	Mean fitness	Maximum fitness	Minimum fitness	Standard deviation
1	0.1507	0.6915	0.0579	0.1317
2	0.1470	**0.6937**	**0.0681**	0.1047
3	0.1491	0.6838	0.0654	0.0975
4	0.1318	0.6788	0.0632	0.1011
5	0.1113	0.6868	0.0579	0.1373
6	0.1490	0.6930	0.0594	0.0963
7	**0.1706**	0.6933	0.0665	**0.1441**
8	0.1473	0.6881	0.0647	0.0970
9	0.1668	0.6912	0.0674	0.1207
10	0.1360	0.6915	0.0580	0.0936

The best (higher) fitness that the genetic algorithm has achieved, for p=3, has been obtained with the solution: (x_1, y_1) = (37.9751; 38.1659), (x_2, y_2) = (60.2136; 75.8572) and (x_3, y_3) = (90.9789; 94.9493). In Figure 3 the best result of the experiments is presented. In case study 1, the best of genetic algorithm obtains a distance total of path of 144.1495. This distance is 98.10% of optimum path without obstacles.

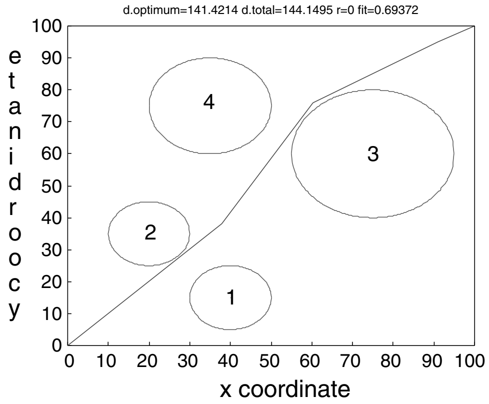


Fig. 3. Best result achieved for the genetic algorithm in case study 1.

As seen above, bacteria colony achieves a better solution than genetic algorithms, but the difference between them is small in simple environments.

4.2 Case study 2: Environment with 12 obstacles

In Table 4 are presented the center positions (x_c, y_c) of the circular obstacles and their respective radius (in meters) for case 2. The results obtained for the bacteria colony are restricted to p=4. In Tables 5 and 6, the results for the case study 2 are summarized.

Table 4. Obstacles for case study 2.

Obstacle nº	Radius	Position (x_c, y_c)
1	05	(50, 50)
2	10	(75, 75)
3	10	(50, 70)
4	05	(20, 20)
5	10	(40, 15)
6	10	(70, 10)
7	08	(65, 40)
8	10	(20, 60)
9	10	(30, 40)
10	08	(85, 50)
11	05	(60, 90)
12	08	(20, 80)

In Figure 4, the best result of the experiments is presented. In this case, the best result of the experiments was: (x_1, y_1) = (85.2011; 70.9190), (x_2, y_2) = (76.2639; 62.0575), (x_3, y_3) = (56.6595; 44.1203) and (x_4, y_4) = (23.09478; 13.0078).

Table 5. Results for an environment with 12 obstacles with bacteria colony.

Experiment	Mean fitness	Maximum fitness	Minimum fitness	Standard deviation
1	0.1405	0.6810	0.0300	0.0830
2	0.1360	**0.6942**	0.0290	0.0808
3	0.1409	0.6866	0.0329	0.0816
4	0.1403	0.6438	0.0331	0.0681
5	0.1274	0.6673	0.0294	**0.0561**
6	0.1424	0.6805	**0.0169**	0.0853
7	**0.1503**	0.6833	0.0294	0.0821
8	0.1416	0.6907	0.0321	0.0710
9	0.1360	**0.6942**	0.0290	0.0808
10	0.1405	0.6810	0.0300	0.0830

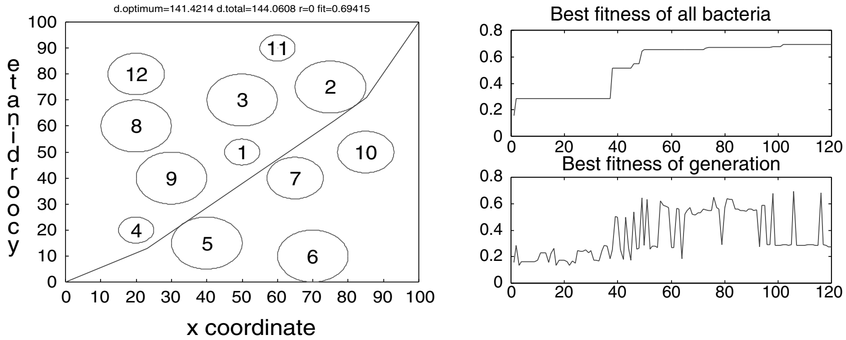


Fig 4. Best result achieved for the bacteria colony in case study 2.

In case study 2, the best of bacteria colony obtains a total distance of 144.0608. This distance is 98.17% of optimum path without obstacles. The average performance would be useful as an indication of the robustness of the configuration of bacteria colony. In Table 5 is observed that the bacterial colony responded well for all the simulations attempts. From 10 repeated simulations, it is shown that the results of bacterial colony were significant for path planning about 120 generations (see Figure 4) for the case study 2.

In Figure 5, the best result of the experiments is presented. In this case, the best result of the experiments was: (x_1, y_1) = (2.8168; 24.3320), (x_2, y_2) = (9.1478; 67.0436), (x_3, y_3) = (88.9937; 93.6584) and (x_4, y_4) = (94.6136; 88.3787).

In case study 2, the best of genetic algorithm obtains a total distance of 172.3573. This distance is 82.05% of optimum path without obstacles. As seen in the comparative study, the robustness of the bacteria colony is higher than the one of the genetic algorithms when dealing with complex environments.

Table 6. Results for an environment with 12 obstacles with genetic algorithm.

Experiment	Mean fitness	Maximum fitness	Minimum fitness	Standard deviation
1	0.0976	0.5560	0.0313	0.0797
2	0.0876	**0.5802**	0.0304	0.0796
3	0.0779	0.5413	0.0368	0.0705
4	0.1036	0.5567	0.0256	**0.1052**
5	0.1010	0.5521	0.0369	0.0828
6	0.0897	0.5462	0.0402	0.0763
7	0.0816	0.5565	0.0352	0.0733
8	**0.1042**	0.5616	**0.0410**	0.1041
9	0.0909	0.5605	0.0396	0.0836
10	0.0890	0.5544	0.0370	0.0757

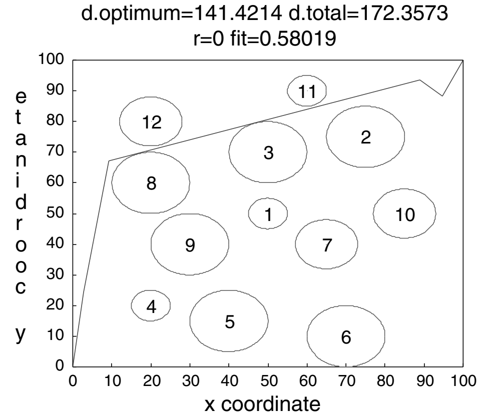


Fig. 5. Best result achieved for the genetic algorithm in case study 2.

5 Conclusion and future works

A research area with special relevance to mobile robot systems is devising suitable methods to plan optimum moving trajectories. There exist many approaches within the area of evolutionary computation and swarm intelligence to solve the problem of optimization of path planning in mobile robotics. In this paper the application of the forage theory of a bacteria colony in form of an optimization algorithm is explored for this purpose.

Must be emphasized that the bacteria own a control system that allows it is foraging capabilities and avoiding noxious substances. In this paper, the possibilities of exploring the bacteria colony efficiency are successfully presented, as shown in two simulated cases study. The results of these simulations are very encouraging and they indicate important contributions to the areas of swarm intelligence and path planning in robotics.

However, in future works, more detailed studies related to population size, number of chemotactic steps, number of swimming steps, number of elimination-dispersal events and movement speed are necessary. In this context, a comparative study of bacteria colony with evolutionary algorithms and others swarm intelligence methodologies will be done.

References

1. Alon, U.; Surette, M.G.; Barkal; Leibler, S. (1999) Robustness in bacterial chemotaxis, *Nature* 397, 14 January 1999, pp. 168-171
2. Baras, J.S.; Tan, X.; Hovareshti P. (2003) Decentralized control of autonomous vehicles, *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, USA, pp. 1532-1537
3. Bennewitz, M.; Burgard, W.; Thrun, S. (2002) Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots, *Robotics and Autonomous Systems* 41, pp. 89-99
4. Bremermann, H.J. (1974) Chemotaxis and optimization, *Journal of Franklin Institute* 297, pp. 397-404
5. Charon, N.W.; Goldstein, S.F. (2002) Genetics of mobility and chemotaxis of a fascinating group of bacteria: the spirochetes, *Annual Review of Genetics* 36, pp. 47-73
6. Dhariwal, A.; Sukhatme, G.S.; Requicha, A.A.G. (2004) Bacterium-inspired robots for environmental monitoring, *Proceedings of the IEEE International Conference on Robotics & Automation*, New Orleans, LA, pp. 1496-1443
7. Dorigo, M.; Di Caro, G. (1999) The ant colony optimization meta-heuristic, in D. Corne, M. Dorigo, and F. Glover (editors), *New Ideas in Optimization,* McGraw-Hill, pp. 11-32
8. Fujimori, A.; Nikiforuk, P.N.; Gupta, M.M. (1997) Adaptive navigation of mobile robots with obstacle avoidance, *IEEE Transactions on Robotics and Automation* 13(4), pp. 596-602

9. Gemeinder, M.; Gerke, M. (2003) GA-based path planning for mobile robot systems employing an active search algorithm, *Applied Soft Computing*, 3, pp. 149-158
10. Kennedy, J.F.; Eberhart, R.C.; Shi, R.C. (2001) *Swarm intelligence*. San Francisco: Morgan Kaufmann Pub
11. Liu, Y.; Passino, K.M. (2004) Stable social foraging swarms in a noisy environment, *IEEE Transactions on Automatic Control* 49(1), pp. 30-44
12. Melchior, P.; Orsoni, B.; Lavaialle, O.; Poty, A.; Oustaloup, A. (2003) Consideration of obstacle danger level in path planning using A* and fast-marching optimization: comparative study, *Signal Processing* 83, pp. 2387-2396
13. Mello, B. A.; Tu, Y. (2003) Perfect and near-perfect adaptation in a model of bacterial chemotaxis, *Biophysical Journal* 84, pp. 2943-2956
14. Müller, S.D.; Marchetto, J.; Airaghi, S.; Koumoutsakos, P. (2002) Optimization based on bacterial chemotaxis, *IEEE Transactions on Evolutionary Computation* 6(1), pp. 16-29
15. Passino, K.M. (2002) Biomimicry of bacterial foraging for distributed optimization and control, *IEEE Control Systems* 22(3), pp. 52-67
16. Silva, L.N.C.; Timmis, J.I. (2002) Artificial immune systems: a new computational intelligence approach, Springer-Verlag, London
17. Tsuji, T.; Tanaka, Y.; Morasso, P.G.; Sanguineti, V.; Kaneko, M. (2002) Bio-mimetic trajectory generation of robots via artificial potential field with time base generator, *IEEE Transactions on Systems, Man and Cybernetics - Part C* 32(4), pp. 426- 439
18. Tu, J.; Yang, S.X. (2003) Genetic algorithm based path planning for a mobile robot, *Proceedings of the IEEE International Conference on Robotics & Automation*, Taipei, Taiwan, pp. 1221-1226
19. Xiao, J.; Michalewicz, Z.; Zhang, L.; Trojanowski, K. (1997) Adaptive evolutionary planner/navigator for robots, *IEEE Transactions on Evolutionary Computation* 1(1), pp. 18-28
20. Ward, M. (2001) BT ponders bacterial intelligence, *BBC News Online Technology*, 13 September 2001

An Empirical Investigation of Optimum Tracking with Evolution Strategies

Karsten Weicker

HTWK Leipzig, IMN, Postfach 301166, 04251 Leipzig, Germany**
weicker@imn.htwk-leipzig.de

Summary. This article reports the results of a thorough empirical examination concerning the parameter settings in optimum tracking with $(1 \overset{+}{,} \lambda)$ evolution strategies. The investigated scenario is derived from real-world applications where the evaluation function is very expensive concerning the computation time. This is modeled by a strong dependence of the dynamics on the number of evaluations. This model enables the derivation of optimal population sizes that might serve as recommendations for general applications. Both random and linear dynamics and two modes for correlating population size and strength of dynamics are examined. The results show for mere tracking problems that plus selection outperforms comma selection and that self-adaptive evolution strategies are able to deliver a close to optimal performance.

Key words: evolutionary computation, evolution strategy, time-dependent optimization

1 Introduction

If optimization problems are considered in real-world applications, one experiences in many cases that the underlying optimization problem is of a dynamic (or time-dependent) nature. Often these problems can be tackled by optimizing a sequence of static problems. However, if this is not possible, the optimization problem changes while the optimization problem proceeds. Then high requirements concerning the optimizer must be met.

Time-dependent optimization problems have been introduced to evolutionary computation by Goldberg and Smith (1987). A survey on problems and techniques can be found in the book of Branke (2002). De Jong (2000) distinguishes four coarse categories of time-dependent problems and gives examples for according real-world applications:

** the work was carried out at the University of Stuttgart, Institute of Formal Methods in Computer Science, Germany

- drifting landscapes where the topology does not change (e.g.control of a chemical production process)
- landscapes undergoing significant morphological changes (e.g. competitive marketplaces)
- landscapes with cyclic patterns (e.g. problems depending on seasonal climate changes)
- landscapes with abrupt and discontinuous changes (e.g. failure of a machine in a production process)

This paper focuses on the first category, the drifting landscapes, where an optimal control point must be tracked. In those tracking problems, only slight modifications are applied frequently to the optimization problem. Then, it is sensible to modify a candidate solution using local operators that create offspring individuals phenotypically close to the parental individual. Within the real-valued domain Gaussian mutation from evolution strategies (Beyer and Schwefel, 2002) are a good choice for such an operator.

We can distinguish dynamic problems with respect to the required promptness of reply. If there are no stringent time requirements the problem can frequently be understood as sequence of static problems as pointed out above. This paper focuses on those problems with severe time restrictions. The restriction is modeled by correlating the offspring population size with the severity of the dynamics. This is a new approach which was only investigated in the related theoretical analysis (Weicker, 2003, 2004). The dependence of the dynamics on the population size is motivated by the fact that in many real-world problems the evaluation function is the most expensive part of the optimization. Therefore, it is sensible that a bigger population size leads to a higher severity (per generation) if the problem is changing continuously.

2 Related Work

In the literature there are basically three different approaches to tackle tracking problems by slight modification of existing candidate solutions. First, for standard binary encoded problems a special variable local search operator has been introduced by Vavak et al. (1996) where a local search phase is started in which first only bits with low order are changed and with missing success the search range is extended by higher order bits. Second, the multinational GA by Ursem (2000) couples the mutation rate and the phenotypic distance of the individual to a special representative of a subpopulation. The subpopulations are clustered phenotypically. Although a bit-flipping operator is used that is not phenotypically local, the control of the mutation rate by a phenotypical distance yields local variation. The third technique are the mutation operators from evolution strategies and evolutionary programming which have been examined on tracking problems since the late 1990s. The Gaussian mutation with an additive Gaussian update rule was investigated by Angeline

(1997). The log-normal update rule was subject of the examinations by Bäck (1997, 1998) and Salomon and Eggenberger (1997). The major focus of these works was on showing the feasibility of tracking. Arnold and Beyer (2002, 2003) analyzed an evolution strategy with global arithmetic recombination theoretically.

Concerning the calibration of the population size within the context of dynamic problems, there is only few research available. Using a Markov model of the search process, Weicker (2003, 2004) examined optimal population sizes for the same problem and the above-mentioned correlation between population size and strength of dynamics.

The standard results of evolution strategy theory (Rechenberg, 1973; Beyer, 2001) are only partially applicable to time-dependent problems since the general conditions change tremendously if the problem depends on the additional time dimension.

3 Methodology

Our goal is to learn more about how an algorithm should be calibrated to enable mere tracking of an optimal control point. As a consequence we consider only the most natural and simple tracking problem, the n-dimensional time-dependent sphere

$$F^{(t)}(x_1, \ldots, x_n) = \sum_{i=1}^{n} (x_i - opt_i^{(t)})^2$$

with the optimum being placed at position $(opt_1^{(t)}, \ldots, opt_n^{(t)})$ at time $t \in \mathbb{N}$. In particular $n \in \{2, 10, 20\}$ are considered in the remainder. The optimum is updated each generation

$$(opt_1^{(t+1)}, \ldots, opt_n^{(t+1)}) = (opt_1^{(t)} + a_1^{(t)}, \ldots, opt_n^{(t)} + a_n^{(t)}).$$

With *random dynamics* random numbers $u_i^{(t)} \in [-1, 1]$ are chosen uniformly and normalized to the strength S of the dynamics

$$a_i^{(t)} = S \cdot \frac{u_i^{(t)}}{\sqrt{\sum_{1 \le k \le n} (u_k^{(t)})^2}}.$$

Note, that the directions towards the corners of the cube $[-1, 1]^n$ have a bigger probability than other directions.

With *linear dynamics* a fixed offset $(a_1^{(0)}, \ldots, a_n^{(0)})$ is created once and used in all generations, i.e. $\forall_{t \in \mathbb{N}_0} : (a_1^{(t)}, \ldots, a_n^{(t)}) = (a_1^{(0)}, \ldots, a_n^{(0)})$. Again the offset is normalized to strength S. In this examination changes are applied only to the first two dimensions ($a_1^{(0)} = \frac{2}{\sqrt{5}} S$, $a_2^{(0)} = \frac{1}{\sqrt{5}} S$, $a_3^{(0)} = \ldots = a_n^{(0)} = 0$).

However this leads to the same results as any other direction of the dynamics since the only variation operator is isotropic.

Since problems with high time-restrictions are the focus of this investigation, the severity S depends on the number of evaluated individuals per generation. The cost α depends on the number of evaluations and cost β comes into effect each time a new generation is processed. With *linear influence* the number of fitness evaluations *evals* has a linear effect on the severity $S = \beta + \alpha \; evals$. This appears to be the natural case. With *squared influence* the number of fitness evaluations *evals* is considered as a squared factor $S = \beta + \alpha \; evals^2$ This scenario models a system which requires a high reactiveness of the control component. We assume $\alpha = 1$ and $\beta \in \{0, 1, 2, 3, 4\}$.

As optimization algorithms $(1, \lambda)$- and $(1 + \lambda)$-evolution strategies with isotropic Gaussian mutation are used. We have run experiments with both the self-adaptive step size adaptation and fixed step size parameter $\sigma = S \cdot \gamma$ where the values

$$\begin{aligned}\gamma \; \in \{ \; & 0.1, 0.125, \ldots, 0.375, 0.4, 0.45, 0.5, 0.55, 0.6, 0.7, \ldots, 3.9, 4.0, \\ & 5.0, 6.0, 8.0, 10.0, 15.0, 20.0, 25.0, 30.0 \; \}.\end{aligned}$$

are used to detect the optimal fixed step size. The experiments using step size adaptation are initialized with $\gamma = 1.0$. The algorithms with fixed step size are included within this examination to approximate the best possible performance. This serves as reference point for the quality assessment of the self-adaptive algorithm. The used offspring population sizes are

$$\lambda \; \in \{ \; 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 25, 30 \; \}.$$

The number of evaluations is $evals = \lambda$ for the comma strategy and $evals = \lambda + 1$ for the plus strategy since the parental individual is evaluated again.

For each algorithm and problem instance 50 independent optimization runs have been executed. In the first generation the individual is placed at the optimum. Each run lasts 80 generations. During the first 30 generations the tracking behavior stabilizes and the last 50 generations are used to assess the tracking performance of the algorithm. The performance is measured as average (fitness) distance of the best individual A to the optimum *opt*.

$$\text{Performance}(Alg) = \frac{1}{50 \cdot runs} \sum_{1 \le i \le runs} \; \sum_{31 \le t \le 80} \left(F^{(t)}(A^{(t)}_{Alg}) - F^{(t)}(opt^{(t)}) \right)$$

4 Results

All results concerning the accuracy and the optimal parameter settings are shown in Table 1.

4.1 Linear dynamics and linear dependence on population size

In the considered problem the optimum moves linearly in a given direction. The offspring population size has a linear impact on the strength of the dynamics. Since the feasibility of tracking is known from theoretical and empirical work (Bäck, 1998; Arnold and Beyer, 2003) we concentrate on the evaluation of the parameter choices.

Comparing the results using the best fixed step size with those using the self-adaptive step size shows that the self-adaptation is competitive within a factor of 3-4. This is due to a higher variance concerning the created offspring individuals which requires in general bigger population sizes. Both factors worsen the accuracy. However, the performance of the self-adaptive evolution strategy is still within an acceptable range if it is compared to suboptimal values for the step size parameter as it is shown in Figure 1. The figure demonstrates that the algorithm with fixed step size is only superior within a small range of step sizes to the self-adaptive version. This confirms the usefulness of the self-adaptive algorithm. Note, that each of the two graphs in Figure 1 is projected to a single value (namely the best accuracy) in Table 1.

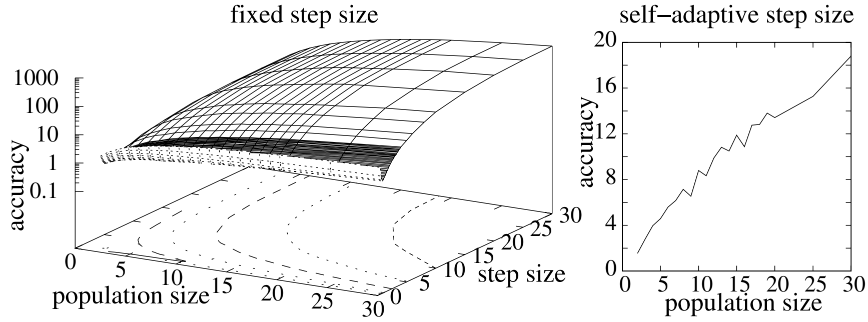


Fig. 1. The averaged accuracy is shown for the problem with linear dynamics, linear dependence on the population size, $\beta = 0$, and dimensionality 20 using plus selection.

Comparing comma and plus strategies in the 2-dimensional case, there is no significant difference concerning the accuracy between both selection schemes (with exception of $\beta = 0$/fixed step size and $\beta = 3$/self-adaptive step size). All cases where significance performance can be found are shown in Table 2. However, in the case of higher dimensions the plus selection scheme is significantly superior to the comma selection. This effect cannot be explained by the known problems of self-adaptation in comma selection with very small offspring population sizes (Schwefel, 1987) since it can be observed too for the algorithm with fixed step size. It is rather due to the fact that in the plus selection the last best approximation delivers a lower bound for the possible worsening of the accuracy.

			linear influence										squared influence									
			fixed step size						self-adaptive				fixed step size						self-adaptive			
			comma			plus			comma		plus		comma			plus			comma		plus	
			acc.	λ	γ	acc.	λ	γ	acc.	λ	acc.	λ	acc.	λ	γ	acc.	λ	γ	acc.	λ	acc.	λ
linear dynamics	2 dim.	$\alpha = 1, \beta = 0$	0.0166	8	1.3	0.0138	2	3.6	0.0671	19	0.0695	14	0.3215	3	2.2	0.0562	2	3.7	6.6062	5	3.2031	2
		$\alpha = 1, \beta = 1$	0.0201	13	1.0	0.0202	9	1.3	0.0752	19	0.0718	19	0.3985	3	2.3	0.0878	2	3.7	7.2549	5	5.0050	2
		$\alpha = 1, \beta = 2$	0.0231	13	1.0	0.0236	14	1.0	0.0820	19	0.0872	18	0.4854	4	1.8	0.1263	2	3.4	7.7493	5	5.7616	5
		$\alpha = 1, \beta = 3$	0.0267	13	1.1	0.0264	13	1.1	0.0955	15	0.0870	25	0.5496	4	1.8	0.1726	2	3.4	8.9979	5	6.0385	5
		$\alpha = 1, \beta = 4$	0.0302	13	1.1	0.0305	14	1.1	0.1022	20	0.0976	25	0.6006	4	1.7	0.2248	2	3.7	9.7085	5	6.8239	5
	10 dim.	$\alpha = 1, \beta = 0$	0.5921	4	1.6	0.2899	2	3.0	2.1939	2	1.1384	2	3.8541	2	1.9	1.1719	2	3.0	8.8694	2	4.6733	2
		$\alpha = 1, \beta = 1$	0.7964	7	1.2	0.6410	3	2.1	2.9702	17	2.5634	2	6.0221	2	1.9	1.8311	2	3.0	13.858	2	7.3021	2
		$\alpha = 1, \beta = 2$	1.0119	7	1.2	0.9392	8	1.2	3.4863	17	3.0671	13	8.6815	2	1.9	2.6376	2	3.0	19.833	2	10.534	2
		$\alpha = 1, \beta = 3$	1.2444	7	1.2	1.1697	8	1.2	3.6841	17	3.5459	16	10.988	3	2.0	3.5864	2	3.0	26.979	2	14.327	2
		$\alpha = 1, \beta = 4$	1.4412	11	1.1	1.3981	9	1.2	4.2589	17	3.9448	13	12.922	3	2.0	4.6877	2	3.0	35.478	2	18.693	2
	20 dim.	$\alpha = 1, \beta = 0$	1.6455	2	1.1	0.8389	2	1.9	2.5036	2	1.5558	2	6.6253	2	1.0	3.4604	2	1.8	9.9441	2	6.0330	2
		$\alpha = 1, \beta = 1$	2.9200	6	1.2	1.8889	2	1.9	5.6359	2	3.5029	2	10.352	2	1.0	5.4069	2	1.8	15.538	2	9.4265	2
		$\alpha = 1, \beta = 2$	3.8383	6	1.2	3.3387	3	1.6	10.014	2	6.2231	2	14.943	2	1.0	7.8027	2	2.1	22.422	2	13.626	2
		$\alpha = 1, \beta = 3$	4.8328	6	1.2	4.6297	8	1.3	15.005	13	9.7236	2	20.327	2	1.0	10.613	2	2.1	30.502	2	18.533	2
		$\alpha = 1, \beta = 4$	5.8035	8	1.2	5.4909	8	1.3	17.020	13	13.761	9	26.501	2	1.0	13.842	2	1.8	39.776	2	24.132	2
random dynamics	2 dim.	$\alpha = 1, \beta = 0$	0.0039	3	0.7	0.0017	2	1.1	0.0155	9	0.0118	4	0.0176	2	0.7	0.0071	2	1.1	0.2114	3	0.0530	2
		$\alpha = 1, \beta = 1$	0.0070	3	0.7	0.0038	2	1.1	0.0205	9	0.0180	8	0.0276	2	0.7	0.0110	2	1.1	0.2610	3	0.0828	2
		$\alpha = 1, \beta = 2$	0.0100	7	0.7	0.0067	2	1.1	0.0222	9	0.0233	8	0.0394	2	0.7	0.0159	2	1.1	0.3190	3	0.1186	2
		$\alpha = 1, \beta = 3$	0.0124	7	0.7	0.0098	3	1.0	0.0275	9	0.0278	8	0.0537	2	0.7	0.0216	2	1.1	0.3758	3	0.1614	2
		$\alpha = 1, \beta = 4$	0.0151	7	0.7	0.0134	3	1.0	0.0315	9	0.0320	8	0.0706	2	0.7	0.0282	2	1.1	0.4166	3	0.2121	2
	10 dim.	$\alpha = 1, \beta = 0$	0.0199	2	0.25	0.0107	2	0.5	0.0519	3	0.0150	2	0.0825	2	0.225	0.0421	2	0.375	0.3232	2	0.0610	2
		$\alpha = 1, \beta = 1$	0.0426	3	0.3	0.0241	2	0.5	0.0922	3	0.0338	2	0.1289	2	0.225	0.0657	2	0.375	0.5050	2	0.0953	2
		$\alpha = 1, \beta = 2$	0.0657	4	0.25	0.0428	2	0.5	0.1281	6	0.0600	2	0.1853	2	0.225	0.0950	2	0.375	0.7272	2	0.1373	2
		$\alpha = 1, \beta = 3$	0.0893	4	0.25	0.0666	3	0.4	0.1653	6	0.0938	2	0.2522	2	0.225	0.1293	2	0.375	0.8930	3	0.1868	2
		$\alpha = 1, \beta = 4$	0.1136	5	0.3	0.0906	3	0.4	0.2001	6	0.1350	2	0.3301	2	0.225	0.1683	2	0.375	1.0482	3	0.2439	2
	20 dim.	$\alpha = 1, \beta = 0$	0.0317	2	0.125	0.0190	2	0.275	0.1309	5	0.0298	2	0.1259	2	0.125	0.0781	2	0.225	0.6236	2	0.1173	2
		$\alpha = 1, \beta = 1$	0.0713	2	0.125	0.0428	2	0.275	0.1872	5	0.0670	2	0.1968	2	0.125	0.1221	2	0.225	0.9743	2	0.1832	2
		$\alpha = 1, \beta = 2$	0.1204	3	0.15	0.0760	2	0.275	0.2570	5	0.1190	2	0.2840	2	0.125	0.1762	2	0.225	1.4036	2	0.2642	2
		$\alpha = 1, \beta = 3$	0.1748	3	0.15	0.1188	2	0.275	0.3328	9	0.1860	2	0.3863	2	0.125	0.2396	2	0.225	1.9102	2	0.3595	2
		$\alpha = 1, \beta = 4$	0.2379	3	0.15	0.1709	2	0.275	0.3978	9	0.2682	2	0.5037	2	0.125	0.3125	2	0.225	2.2598	3	0.4690	2

Table 1. Best achieved results for random/linear dynamics and linear/squared influence of the offspring population size on the strength of dynamics. The average tracking accuracy is shown together with the optimal population size λ and the optimal value of the step size parameter γ in case of the fixed step size.

		linear dynamics				random dynamics			
		lin. inf.		squ. inf.		lin. inf.		squ. inf.	
		fixed	self-ad.	fixed	self-ad.	fixed	self-ad.	fixed	self-ad.
2 dim.	$\alpha = 1,\ \beta = 0$	+		+	+		+	+	+
	$\alpha = 1,\ \beta = 1$			+	+		+	+	+
	$\alpha = 1,\ \beta = 2$			+	+		+	+	+
	$\alpha = 1,\ \beta = 3$		+	+	+	+		+	+
	$\alpha = 1,\ \beta = 4$			+	+	+		+	+
10 dim.	$\alpha = 1,\ \beta = 0$	+	+	+	+			+	+
	$\alpha = 1,\ \beta = 1$	+	+	+	+	+	+	+	+
	$\alpha = 1,\ \beta = 2$	+	+	+	+	+	+	+	+
	$\alpha = 1,\ \beta = 3$	+	+	+	+	+	+	+	+
	$\alpha = 1,\ \beta = 4$	+	+	+	+	+	+	+	+
20 dim.	$\alpha = 1,\ \beta = 0$	+	+	+	+		+	+	+
	$\alpha = 1,\ \beta = 1$	+	+	+	+	+	+	+	+
	$\alpha = 1,\ \beta = 2$	+	+	+	+	+	+	+	+
	$\alpha = 1,\ \beta = 3$	+	+	+	+	+	+	+	+
	$\alpha = 1,\ \beta = 4$	+	+	+	+	+	+	+	+

Table 2. Cases where the plus selection scheme is significantly better than the comma selection.

With decreasing influence of the population size on the severity (which is modeled by an increasing β), the accuracy values of comma and plus strategies are getting very close quantitatively – although they are still significantly different with dimensions 10 and 20.

Concerning the population size, bigger values are optimal with increasing β. Also the plus strategy has a tendency to require smaller population sizes than the comma strategy. This is a second factor contributing to its superior performance.

The optimal step size values are in most cases bigger with small values of β. Comparing again comma and plus strategies, in all cases the plus strategy uses a step size that is bigger (or equal) to the step size of the comma strategy. This is probably due to the lower bound of the worsening and accounts for the better performance since bigger improvements are more likely. Decreasing step size values for increasing dimensionality is known from results by Rechenberg (1973).

4.2 Linear dynamics and squared dependence on population size

If the linear influence of the offspring population size on the dynamics is replaced by a squared influence, the accuracy deteriorates considerably which is natural.

The difference in the accuracy is more obvious between comma and plus selection than in the previous section. The plus selection is significantly better in all considered cases (see Table 2).

As one would expect, the optimal offspring population size is considerably smaller than with a linear influence. Only for the comma selection, population sizes different from 2 are found to be optimal. This is the case with dimension 2 and with dimension 10 combined with bigger values for β.

Again the plus selection requires bigger step size values than the comma selection. Also we observe that the squared dependence of the severity on the population size leads to bigger step size values in almost all cases.

4.3 Random dynamics

In the case of dynamics with randomly changing directions and a linear influence of the offspring population size onto the severity of the dynamics, the effective movement of the optimum is considerably slower. As a consequence the tracking accuracy is better compared to the linear dynamics and both smaller population sizes and step sizes are necessary. Also we could observe in the problems with linear dynamics an enormous worsening of the tracking accuracy with increasing dimensionality – this is not the case with random dynamics anymore. Presumably this is also due to the slower effective movement of the optimum and the resulting smaller step sizes.

In addition, we observe that plus selection is again significantly better than the comma selection in almost all cases. Exceptions are shown in Table 2. Again $\beta = 0$ as well as dimensionality 2 seem to be less favorable for the plus selection. Again plus selection requires smaller population sizes and bigger step sizes.

Another very interesting conclusion can be drawn from the ratio of the accuracy of the self-adaptive ES to the accuracy of the ES with fixed step size. Although the self-adaptive ES has a worse accuracy for both linear and random dynamics, this ratio is considerably smaller if random dynamics are involved. Apparently the self-adaptation mechanism is able to soften irregularities in the dynamics due to the random movement. This is an advantage over the algorithm using the fixed step size. As a consequence the performance of the plus strategy with self-adaptation comes very close to the best performance using a fixed step size.

For random dynamics and the severity depending on the offspring population size to the square, analogous observations hold as in Section 4.2. Moreover, the population size needs to be even smaller in this scenario: population size 2 is best and only comma selection in the self-adaptive cases may lead to population size 3. Within the considered cases, the optimal step sizes are constant for each dimensionality. With increasing dimensions the optimal step size decreases.

5 Conclusion

The considered problem is difficult to analyze theoretically because of the linkage between population size and the strength of the dynamics. However, we believe that this model reflects the requirements of many real-world applications.

For the mere tracking problem with a good-natured, isotropic fitness landscape around the optimum, the paper gives concrete recommendations how an evolution strategy should be parameterized to achieve best tracking accuracy. These results are independent of the severity of the problem and generally applicable. In fact, the considered scenario allows the derivation of optimal population sizes. The most important recommendations are summarized.

- Plus selection is preferable over comma selection if self-adaptation is used but also with fixed step sizes and dimensionality > 2.
- Plus selection requires smaller population size and bigger step size which is primarily due to the nature of optimum tracking where a good approximation of the previous generation is still very close to the optimum. This effect supplies a better worst case approximation with less evaluations. Concerning the plus selection, $\lambda = 1$ should also be considered at least for higher dimensionality since this might improve the tracking accuracy even more.
- The optimal population size decreases with increasing dimensionality in accordance to the results of Rechenberg (1973) for static problems.

Also the empirical examination could confirm the results of Weicker (2003, 2004) for comma selection and a discrete problem with low dimensionality.

All in all, we showed that self-adaptive evolution strategies are highly competitive and perform close to the performance using the best possible fixed step size. This is especially true if problems with random dynamics are tackled.

References

Peter J. Angeline. Tracking extrema in dynamic environments. In Peter J. Angeline, Robert G. Reynolds, John R. McDonnell, and Russ Eberhart, editors, *Evolutionary Programming VI*, pages 335–345, Berlin, 1997. Springer. Lecture Notes in Computer Science 1213.

Dirk V. Arnold and Hans-Georg Beyer. Random dynamics optimum tracking with evolution strategies. In Juan Julián Merelo Guervós, Panagiotis Adamidis, Hans-Georg Beyer, José-Luis Fernández-Villacañas, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature – PPSN VII*, pages 3–12, Berlin, 2002. Springer.

Dirk V. Arnold and Hans-Georg Beyer. Optimum tracking with evolution strategies. Technical Report CI 143/03, Universität Dortmund, Dortmund, 2003.

Thomas Bäck. Self-adaptation. In Thomas Bäck, David B. Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, pages

C7.1:1–15. Institute of Physics Publishing and Oxford University Press, Bristol, New York, 1997.

Thomas Bäck. On the behavior of evolutionary algorithms in dynamic environments. In *IEEE Int. Conf. on Evolutionary Computation*, pages 446–451. IEEE Press, 1998.

Hans-Georg Beyer. *The theory of Evolution Strategies.* Springer, Berlin, 2001.

Hans-Georg Beyer and Hans-Paul Schwefel. Evolution strategies: a comprehensive introduction. *Natural Computing*, 1:3–52, 2002.

Jürgen Branke. *Evolutionary Optimization in Dynamic Environments.* Kluwer, Boston, 2002.

Kenneth De Jong. Evolving in a changing world. In Z. Ras and A. Skowron, editors, *Foundation of Intelligent Systems*, pages 513–519, Berlin, 2000. Springer.

David E. Goldberg and Robert E. Smith. Nonstationary function optimization using genetic algorithms with dominance and diploidy. In J. J. Grefenstette, editor, *Proc. of the Second Int. Conf. on Genetic Algorithms*, pages 59–68, Hillsdale, NJ, 1987. Lawrence Erlbaum Associates.

Ingo Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* frommann-holzbog, Stuttgart, 1973. German.

Ralf Salomon and Peter Eggenberger. Adaptation on the evolutionary time scale: A working hypothesis and basic experiments. In J.-K. Hao, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyders, editors, *Artificial Evolution: Third European Conf., AE'97*, pages 251–262, Berlin, 1997. Springer.

Hans-Paul Schwefel. Collective phenomena in evolutionary systems. In P. Checkland and I. Kiss, editors, *Problems of Constancy and Change – The Complementarity of Systems Approaches to Complexity*, pages 1025–1032, Budapest, 1987. Int. Institute for Applied Systems Analysis (HASA).

Rasmus K. Ursem. Multinational GAs: Multimodal optimization techniques in dynamic environments. In Darrell Whitley, David Goldberg, Erick Cantu-Paz, Lee Spector, Ian Parmee, and Hans-Georg Beyer, editors, *Proc. of the Genetic and Evolutionary Computation Conf. (GECCO-00)*, pages 19–26, San Francisco, CA, 2000. Morgan Kaufmann.

Frank Vavak, Terence C. Fogarty, and K. Jukes. A genetic algorithm with variable range of local search for tracking changing environments. In Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature – PPSN IV*, pages 376–385, Berlin, 1996. Springer.

Karsten Weicker. *Evolutionary Algorithms and Dynamic Optimization Problems.* Der andere Verlag, Osnabrück, Germany, 2003.

Karsten Weicker. Analysis of local operators applied to discrete tracking problems. to appear in Softcomputing Journal, 2004.

Implementing a Warning System for Stromboli Volcano

G. Nunnari(1), G. Puglisi(2) and F. Gargaun(3)

(1) Dipartimento di Ingegneria Elettrica, Elettronica e dei Sistemi,
Università di Catania, Viale A. Doria, 6, 95125 Catania, Italy
gnunnari@diees.unict.it
(2) Istituto Nazionale di Geofisica e Vulcanologia, Sez. di Catania, Piazza
Roma, 2, 95125 Catania, Italy,
puglisi-g@ct.ingv.it
(3) Department of Signal Circuits and Systems, The "Gh. Asachi" Technical
University of Iasi, Bvd. Carol 11, Iasi 700050, Romania,
radu@altastiinta.ro

Abstract – In this paper we describe a particular warning system for the purpose of monitoring ground deformation at Stromboli volcano (Aeolian Islands, Sicily). The paper, after a short introduction about the features of the recording system, describes the solutions adopted for modeling both short and long range ground deformation episodes. The guide lines for implementing a fuzzy rule base for the aims of the warning system are also outlined.

Keywords: warning systems, fuzzy logic, neural networks, ground deformation, volcanoes

1 Introduction

Despite equipments for monitoring active volcanoes are widely used (see for instance McGuire et al., 1995), automatic system for evaluating measured data are still lacking. Mathematical models to fit observed data are quite complicate and do not allow to cover all needs. In this framework some benefit can derive from the use of Soft-computing techniques since they allow a fusion of models and heuristic knowledge (Fortuna et al., 2001). This paper deals with an attempt to implement a warning system for evaluating episodes of ground deformation recorded at Stromboli volcano.

The surface of Stromboli, such as all others active volcanoes, deforms when magma moves beneath it or rises into its conduits. Hundreds of shallow cracks or deep faults tens to hundreds of meters long may develop in hours or days. The ground can change shape by rising up, subsiding, tilting, or forming fractures that are clearly visible. A variety of methods can be considered to monitor ground deformation in active volcanic areas such as EDM (Electronic Distance Measuring) or satellite based techniques such as the GPS (Global Positioning Systems) or SAR (Synthetic Aperture Radar). Despite of the use of these modern technologies measuring and evaluating ground deformation data is not a completely automatic process. Indeed, for instance, traditional EDM techniques require a massive use of human operators to reach the summit of a volcano in order to perform the measuring process. However, at the present day's remote control instrumentation is commercial available in order to make automatic measuring. Such a kind of measuring systems consists of a remote control sensor that can be programmed to measure slope distances and angles among the sensor and appropriate reflectors positioned at ground level. Recorded information are then sent to a central recording centre (e.g. an observatory) by using an appropriate communication link (e.g. radio or/and cable). The use of such a kind of equipments is important in order to have real-time information about the state of ground deformation and represents the premise to reduce the risk for people living around active volcanic areas.

However the availability of information in real time, even necessary, is not enough to implement a system able to automatically evaluate the state of ground deformation and eventually to issue alarm messages. To reach a similar goal, it is necessary to implement appropriate software procedures able to process and interpret ground deformation data. The main aim of this paper is to present a warning system, currently under developing at the INGV, to process ground deformation data recorded in the Stromboli volcano (Aeolian island, Sicily) by an automatic measuring system, referred here as THEODOROS. The developing of such a systems was triggered by the recent volcanic activity at Stromboli, culminated with a tsunami on December 30th, 2002, due to the landslide of a large volume of ground material (17 million cubic meters) in the surrounding sea. This event was very dangerous for people living in this area that was evacuated for a few weeks.

2 The Stromboli Volcanic Areas and the THEODORO System

Stromboli is an active volcano, about 2500 m high above the sea floor. Roughly speaking only the last Kilometre of this volcano emerges from the sea, forming an island whose diameter ranges from 2.4 to 5 km. It belongs to the Aeolian Islands and represents the most active volcano of this archipelago. The volcanic hazard of Stromboli is mainly characterized by explosive activity (Bertagnini et al. 1999) and in particular by paroxysmal episodes that, for example, killed ten people and injured dozens in 1919 and 1930. In the last century, ballistic ejects emitted during these paroxysms destroyed many buildings and fires induced by scoria caused severe damage to crops. During some paroxystic explosions, tsunamis have been signalled (6 times since 1879). Although chronicles of these events do not report any specific origin of these phenomena, Barberi et al. (1993) relate it to hot avalanches or pyroclastic flows entering into the sea during the paroxysm. Since 1993, Stromboli ground deformations have been monitored by GPS and clinometric techniques. This monitoring system was able to detect magmatic intrusions as, for instance, in 1995 (Bonaccorso, 1998). The Stromboli 2003 volcanic emergency prompted setting up a powerful and complex system for monitoring ground deformations both in the SdF area and over island as a whole, based on the integration of different surveying techniques: topography (first classic and then automatic-robotized) and GPS.

2.1 The THEODORO Measuring System

The set up of this new system was a real challenge, given the logistic and operative conditions existing on Stromboli during the volcanic crisis. The chosen instrument was the Leica TCA 2003 (Leica and Geodetics, 2002) Total Station (TS) equipped with GeoMos software that allows remote control of the sensor. The acronym of this system operating at Stromboli is THEODOROS (THEOdolite and Distance-meter Robot Observatory of Stromboli). To guarantee a continuous stream of data from the instrument, it requires a continuous power supply and a continuous link with the PC that controls the Total Station's activities, installed on the S. Vincenzo Observatory, where the National Department of Civil Protection (DPC) control room is located. These technical solutions were achieved by exploiting two facilities already installed by the DPC and the Istituto Nazionale di Geofisica e Vulcanologia (INGV). A reference system of 5 reflectors (Fig. 1), (400, CURV, BORD, SEMF, SPLB2) was installed around the TS, on sites considered stable form the geological point of view. The TS it aimed at guaranteeing the same orientation in the space (both horizontally and vertically) through the measurements of appropriate reference baselines. At the very beginning THEODOROS carried out the measurements only on four reflectors installed on the SdF (SDF5, SDF6, SDF7 and SDF8), that were the only remaining ones after the destruction due to the lava flows, and on three reflectors installed around the site belonging to the reference system (SEMF, 400, SPLB2). After the end of the eruption the network was improved by setting up other reflectors in the SdF and redefining the reference network. The actual system aims to follow the post-eruptive movements of the recent lava flow field. At present, THEODOROS consist of 5 reflectors for the reference system (400, BORD, SEMF, SPLB2 plus ELIS), 10 reflectors for monitoring movements in the SdF (the four previously installed plus SDF9, SDF10, SDF11,SDF12, SDF14, SDF16) and 2 more reflectors to check the stability of the measurement both on short and very long distance measurements (CURV and CRV) (Fig. 2.1). The current measurement strategy assumes that the horizontal geometry and the relative distance of the reference system are fixed (by fixing the coordinates of the reference system benchmarks), while the TS adjusts its own vertical positions by using only the internal vertical compensators.

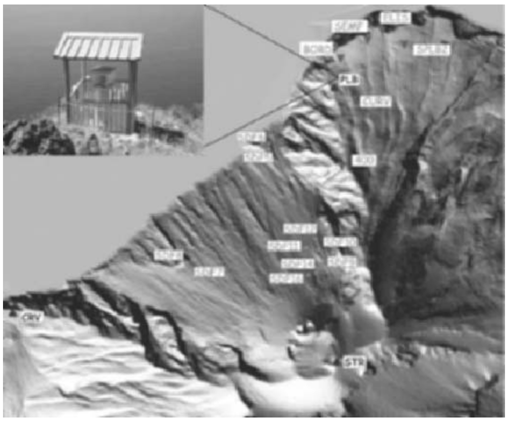


Fig. 2.1. The position of the reflectors and the total station.

3 Data Analysis

In this section we describe some preliminary analysis with the purpose of characterizing some features that could be useful in order to implement a warning system for THEODOROS. We process the time series referring the most interesting points (SDF 9, SDF10, SDF11, SDF14 and SDF16), i.e. the points which exhibits the more relevant movements. For each measuring station relevant information is represented by the slope distance (*sd*), the horizontal (*hz*) angle and the vertical angle (*ve*). Starting from these information the Leica TCA 2003 system is able to transform a vector whose component are *sd, hz, ve* into an equivalent vector whose component are expressed in terms of North (N), South(S) and Quota (Q). As usual, measurements are affected by some kinds of noise sources. In particular, in the considered case it is possible to recognise two main sources of noise: a) thermoelastic effects which produces daily and seasonal deformations of the measuring site. Such a kind of ground deformation is induced by thermal variation related with earth rotation (around its own axes and around the sun), b) noise induced by changes in local atmospheric conditions which affect the propagation of the laser beam considered for distance measuring.

The spectra of a generic slope distance time series, performed by using the Fast Fourier Transform (FFT) algorithm pointed out that the signal is affected by a marked daily (i.e. 1 day) component. Furthermore, time series are typical affected by an evident trend effect, how it appears from Fig.3.1, where the slope distance referred to the SDF 9 measuring point are shown.

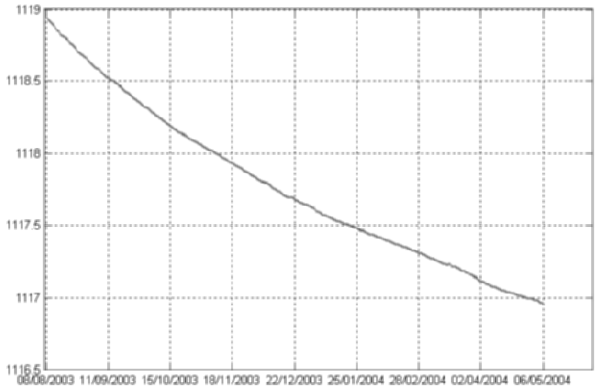


Fig. 3.1. The slope distance at SDF 9 from August 2003 to May 2004. The scale in ordinate is in meters.

Similar behaviour is observed also for horizontal and vertical angles. In order to implement a warning system we need to characterize both diurnal and trends effects during what we will refer to as *normal* activity, i.e.

during quiescent time period of the volcano. To characterise normal activity we have introduced two quantities, referred to as *DMVSD* and *LRVASD*. The first quantity (1) represents the Daily Maximum Variation of the Slope Distance *(DMVSD)* computed over the whole available data set, while the second (2) represents the Maximum Prediction Error of the Slope Distance *(MPESD)*:

$$DMVSD = \max_{t}[\max(sd(t)) - \min(sd(t))] \tag{1}$$

$$MPEASD = \max_{t}[(asd(t)) - pasd(t))] \tag{2}$$

In expressions (1) and (2) *t* represents the generic day during the whole data set available of quiescent activity, *asd(t)* and *pasd(t)* are the observed and estimated average slope distance at day *t.*

The quantity *DMVSD* can be is useful to detect rapid changing phenomena such as magma intrusions and/or suddenly failure of the landslide equilibrium that could represent new hazard situations. Indeed, once a reliable estimation of *DMVSD* has been obtained for each measured baseline, during a representative time period of normal activity, one can compare this value with the max-min difference computed during the previous day. We have found that *DMVSE* is not affected by trend effects so that its variability should be attributed to phenomena related to the daily earth rotation. We expect that when some volcanic phenomena will suddenly occur (e.g. fractures or suddenly landslide) a significant difference between the measured max-min slope distance and the value of *DMVSD* estimated during quiescent activity will be observed. If this is the case, this information will be carefully evaluated together with analogous information related to others baseline and eventually an alarm message might be issued.

The quantity *MPEASD* represents the maximum expected prediction error of the daily average value of the slope distance during a time period of quiescent activity. Experimental data show that during time period of quiescent activity, the variability of the measured quantities (e.g. slope distances) can be easily predicted with high accuracy even with simple methods such as a polynomial approximation or a simple NAR model, as shown in the next section. Thus we can estimate well in advance, (say weeks) the expected value of the daily average slope distance for a given baseline (see the example reported in the next section). Thus if at the current day *t* we find that the difference between the predicted daily average distance *pasd(t)* made in the past, say 1 week before, for a given base line is significantly different, compared with the *MPEASD* evaluated on a representative time period of quiescent activity, then this information should carefully evaluated, together with other similar information of different baselines and, eventually, an alarm message might be issued. The main difference between *DMVSD* and *MPEASD* is that the first quantity can be useful to evaluate ground deformation phenomena suddenly occurring (1 day) while the second can detect long term ground deformation, i.e. episodes that progress during weeks, such as slow and continuous sliding of the SdF.

3.1 Estimating DMVSD

According with expression (1) in order to estimate *DMVSD* we have considered for each baseline the difference between the maximum and minimum value of the slope distance over the available data set. Despite the presence of trend effects the max-min difference shows a clear upper bound that we have referred as *DMVSD* that, for SDF9 is about 2.5 cm. This upper bound can be usefully considered to characterize the significance of short term variation. In other words the estimation of *DMVSD* during the absence of volcanic activity allows detecting when short term variations in the slope distance may be attributed to the presence of a volcanic source. For instance we can say that if the diurnal variation of the slope distance is larger that 2 or 3 times *DMVSD,* something is occurring due the volcanic source. For all the stations there is a clear upper-bound that characterize the movement of the point. These values are used, among other information, as the inputs for the warning system.

3.1 Estimating MPEASD

In order to compute *MPEASD* it is necessary to estimate the daily average value of the slope distance over the whole available data set. This can be done in several ways. Here we have considered two different approaches: the first is a polynomial approximation approach while the second make use of a NAR (Non-linear Auto Regressive) model identified by using a Multi-layer Perceptron neural network (Narendra and Parthasarathy, 1990). The polynomial approach consists in modelling the average daily distance by using the following expression:

$$pasd(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2 + \cdots + a_n \cdot t^n \tag{3}$$

The unknown coefficients a_0, a_1, … a_n can be obtained by using the Least Square algorithm (see for instance Lyung and Soderstrom, 1983). As an example it was estimated that *MPEASD,* for the considered baseline is about 2 cm. The second approach provided similar results.

4 Implementing a Warning System for THEODOROS

The warning system for THEODOROS is based on the availability, for each baseline, of estimation for the quantities *DMVSD* and *MPEASD*. In particular the estimation of *MPEASD* allows classifying network points in different sets. Each set contains points which are characterised by similar *MPEASD.* In our system we have recognised two sets of points (Puglisi et al., 2004). One set is represented by points SDF9, SDF10 and SDF11 in Figure 4.1, which present homogeneous variations of the average slope distance and are rapidly moving with respect to the others. To simplify we describe below the strategy adopted by the warning system referring to *MPEASD* only. Based on the availability of these parameters we have tried to build a warning system, i.e. a system that is able to issue appropriate attention or alarm messages when the predicted daily average slope distance variation exceeds, in a fuzzy sense, the value of *MPEASD* estimated for the given baseline. In more detail we evaluate a variable, referred here as *AI* (Alarm Intensity) that represents the strength of the volcanic sources with respect the daily average slope distance variation. A fuzzy approach was adopted to evaluate *AI* based on values assumed by *MPEASD*. Three fuzzy sets were considered to evaluate the fuzzy variable *AI* referred below as *normal*, *pre-alarm* and *extra-alarm*. The range of variation for *AI* was conventionally assumed to be [0, 100] (Fig. 4.1a).

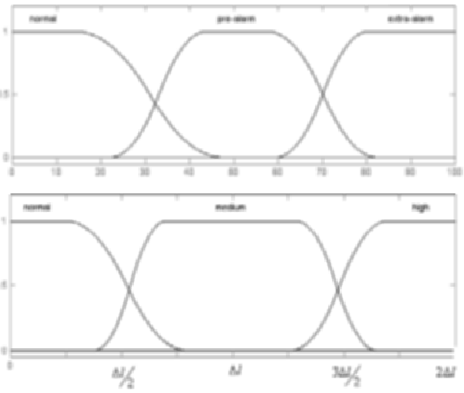


Fig. 4.1. a) (Upper) fuzzy sets for AI, b) (bottom) Fuzzy sets for I

Furthermore, let us indicate as •*I* the difference between the observed daily average slope distance and the corresponding maximum expected prediction error *MPEASD.* Since •*I* will be processed in fuzzy terms, it has been fuzzyfied as indicated in Fig. 4.1b). In the simple case of a warning system based on measures performed on an individual baseline (say for instance SDF9) we can image a trivial rule base of the form:

R1: IF min-max_SDF9 is *normal* THEN Alarm Intensity is *normal*
R2: IF min-max_SDF9 is *medium* THEN Alarm Intensity is *pre-alarm*
R3: IF min-max_SDF9 is *high* THEN Alarm Intensity is *alarm*

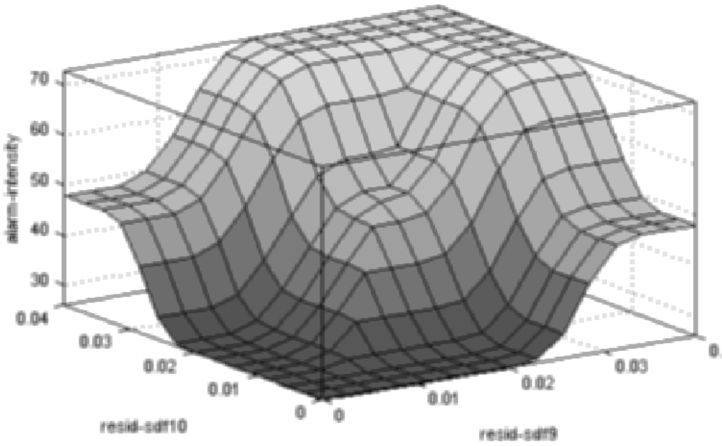


Fig. 4.2. Decision curve considering two clusters of points.

Unfortunately the real case is not so simple. Indeed it is easy to understand that decision cannot be taken based on measures carried out on an individual baseline (e.g. SDF9) but on the whole available data set. At the present, according with analysis so-far carried out, we have implemented a rule base based on the assumption that the whole set of measuring points can be classified into two sets according with rate of land sliding (Puglisi et al. (2004)). This allowed building a rule base that generates a decision surface such that shown in Fig. 4.2. However the warning system is still under developing and its reliability has not tested on significant ground deformation episodes. Work is in progress to this end

5 Conclusions

In this paper some preliminary results obtained with the aim of implementing a warning system for evaluating ground deformations that occur at Stromboli volcano have been presented. Analysis carried out on the data set so-far available suggests to introduce two quantities, namely *DMVSD* and *MPEASD* to evaluate both short and long term ground deformation phenomena. One of the peculiarities of the work is the strategy implemented to set up the warning system. Indeed, since there are not mathematical models to evaluate information carried out trough the measuring process, we have decided to adopt a fuzzy strategy heavily based both on evidences emerged from data analysis and knowledge from human experts. Work is still in its experimental phase and several improvements are needed. Furthermore a testing phase on relevant ground deformation episodes is still lacking due to the limitation imposed by the actual data set.

References

Barberi F, Rosi M and Sodi A (1993) Volcanic hazard assessment at Stromboli based on a review of historical data. Acta Vulcanologica 3: 173-187.

Bertagnini A, Coltelli M, Landi P, Pompilio M, Rosi M (1999) Violent explosions yield new insights into dynamics of Stromboli volcano. EOS Transactions of the American Geophysical Union 80: 633-636.

Bonaccorso A. (1998): Evidence of a dyke-sheet intrusion at Stromboli volcano inferred through continuous tilt. Geophys. Res. Lett., 25 (22), 4225.

Narendra K.S., Parthasarathy, 1990, Identification and Control of dynamical systems using neural networks, vol. 1, pp. 4-27.

Fortuna L, Rizzotto G, Lavorgna M, Nunnari G, Xibilia M.G, Caponetto R. (2001), Soft Computing. New Trends and Application, Springer.

Leica and Geodetics Inc. (2002), Software Crnet user manual.

McGuire B., Kilburn C., Murray J (Editors), (1995) Monitoring active volcanoes – strategies, procedure and techniques, UCL press.

Lyung L, Soderstom T., (1983), Theory and Practice of Recursive Identification, Prentice Hall.

Puglisi G., Bonaccorso A., Mattia M., Aloisi ,M., Bonforte A., Campisi O., Cantarero M., Falzone G., Puglisi B., Rossi M., (2004) New integrated geodetic monitoring system at Stromboli volcano (Italy), submitted to Engineering Geology.

Part VI

Scheduling and Layout

A Genetic Algorithm with a Quasi-local Search for the Job Shop Problem with Recirculation

José António Oliveira

Dept. Produção e Sistemas, Escola Engenharia da Universidade do Minho
Braga, Portugal
zan@dps.uminho.pt

Abstract. In this work we present a genetic algorithm for the job shop problem with recirculation. The genetic algorithm includes a local search procedure that is implemented as a genetic operator. This strategy differs from the memetic algorithm because it is not guaranteed that the local minimum is achieved in each iteration.

This algorithm performs well with classical JSP and can be applied to the JSP with recirculation. The genetic algorithm includes specific knowledge of the problem to improve its efficiency.

We test the algorithm by using some benchmark problems and present computational results.

Keywords: Metaheuristics, Job Shop Scheduling, Genetic Algorithm

1 Introduction

In the job shop problem (JSP) each job is formed by a set of operations that has to be processed in a set of machines. Each job has a technological definition that determines a specific order to process the job's operations, and it is also necessary to guarantee that there is no overlap, in time, in the processing of operations in the same machine; and that there is no overlap, in time, in the processing of operations of the same job. The objective of the JSP is to conclude the processing of all jobs as soon as possible, this is, to minimize the makespan, $C_{\max}$.

The JSP represents several real situations of planning and for that it is a very important problem. Recently, the load operations in a warehouse were modeled by a JSP with recirculation (Oliveira 2001).

Along the years it was made a lot of investigation in this problem, particularly with genetic algorithms. An important issue in the genetic algorithms is the efficiency. In this paper it is presented the inclusion of a "*new type*" of local research to improve the efficiency of the genetic algorithm.

The paper is organized in the following way: in the next section, is described the job shop problem; in the third section is dedicated to the characterization of the adopted solution's methodology; the fourth section presents computational results; and at the end, the conclusions about the work are discussed.

2 Classical Job Shop and Job Shop with Recirculation

In the simplest of cases, the JSP considers a set of jobs, $\{J_1, J_2, ..., J_n\}$ and a set of machines, $\{M_1, M_2, ..., M_m\}$. Each job J_i, consists in a set, of operations, $\{o_{i1}, o_{i2}, ..., o_{im}\}$, among which precedence relations exist that define a single order of processing. Each operation o_i is processed in one machine only, during p_i units of time. The instant when job J_i is concluded is C_i.

For the convenience of the representation, operations are numbered consecutively from o_1 to o_N, in which *N* is the total number of operations. The classical model considers that all the jobs are processed once in every machine, and the total number of operations is $N = n \cdot m$. In a more general model, a job can have a number of operations different from *m* (number of machines). In a case in which a job is processed more than once in the same machine, this is called a job shop model with recirculation.

2.1 Disjunctive Graph

Roy and Sussman (1964) modulated the JSP using a graph, $G = (V, C \cup D)$, which they designated as disjunctive graph. The vertices set, *V*, is constituted by the set of all of the jobs operations $O = \{o_1, o_2, ..., o_N\}$, and by two other fictitious operations, o_0 and o_{N+1} which represent, respectively, on the graph a source and a terminal, $V = \{o_0, o_1, o_2, ..., o_N, o_{N+1}\}$. The operation o_0 represents the beginning of the schedule and precedes all operations. The operation o_{N+1} represents the end of the schedule and succeeds every operation. To each vertex o_i is associated the processing time p_i. The vertices o_0 and o_{N+1} have a null processing time.

The arc set is constituted by one *C* set of conjunctive arcs (oriented arcs) and by a *D* set of disjunctive arcs (non-oriented arcs). In set *C* there is an arc for each ordered pair of operations (o_i, o_j) related by the precedence restrictions $o_i \prec o_j$ that, to represent the precedence relation, $o_i \prec o_j$, is oriented from *i* to *j*. These arcs represent the technological definition of processing of the jobs, which is to say, they define the processing operations order that belong to the same job.

In set *D* there is a non-oriented arc for each pair of operations (o_i, o_j) that must be performed by the same machine. In the model with recirculation, the disjunctive arc that joins operations of the same job, which are processed in the same machine, makes no sense and can be removed from the model, being replaced by the already existing conjunctive arcs, which define the precedence relations between operations of the same jobs. In this way, there will only be disjunctive arcs between operations that are processed by the same machine and that belong to different jobs.

The sequence principle on the disjunctive graph is the attribution of a direction to the disjunctive arcs, that is to say, it is the definition of a processing order between all operations that are processed by that same machine. A schedule is valid if the resulting oriented graph is acyclic. The longest path length is also designated by a critical path between vertices o_0 and o_{N+1} of the acyclic graph and is equal to the value of $C_{\max}$.

2.2 Complexity and Types of Solutions

The JSP is NP-hard (Garey et al. 1976). Even the problem with unitary processing times is minimal NP-hard for three or more machines $J3 \left| p_{ij} = 1 \right| C_{\max}$ (Brucker and Knust 2003, Lenstra and Rinnooy Kan 1979). The exact solution methods are not yet considered efficient to solve large-dimension instances. It is considered that the instances 15×15 still constitute a limit for the exact solution methods (Aiex et al. 2003, Binato et al. 2001).

Much research has been focused on obtaining and improving solutions for the JSP. For a review and comparison, we refer the reader to Jain and Meeran (1999), and to Vaessens et al. (1996).

The solutions (schedules) for the JSP can be classified in 3 sets: semi-active, active, non-delayed. These solutions obey to relations of inclusion that are illustrated in the Venn diagram that is shown in the Figure 1. In an active solution the processing sequence is such that no operation can be started any earlier without either delaying some other operation or violating the precedence restrictions. In a non-delay solution no machine is kept idle when it could start processing some operation (French 1982).

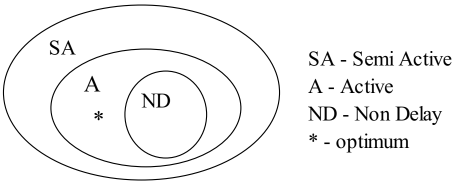


Figure 1. Type of schedules

In relation to the optimal solution of the problem (minimization of $C_{\max}$), we know that it is an active schedule but not necessarily a non-delayed schedule.

2.3 Neighbourhood Schemes

The JSP solution consists in obtaining an orientation of the disjunctive arcs in such way that we can achieve the minimum critical path. A critical block of operations is defined as a set with the maximum successive operations that belong to the critical path and that are processed in that same machine. In literature are introduced some neighbourhood schemes based on the reorientation of some disjunctive arcs of the critical path. Among the already existing literature we emphasize the scheme proposed by Nowicki and Smutnicki (1996) because it is simple and limitates the exchanges to the operations of extremities of the blocks.

Example 1

Consider a preparation of five jobs. The jobs' operations are preformed in one of four machines. All operations o_i have $p_i = 1$. Table 1 defines the operations' sequence for each of the jobs, identifying the machine where the operation is processed.

Table 1. Identification of machine for each operation

	i^{th} operation				
Job i	1st	2nd	3rd	4th	5th
1	1	1	3	1	2
2	2	2	2	1	2
3	3	3	3	1	3
4	4	1	3	4	3
5	2	4	4	4	3

Consider the following numbering of the operations, in agreement with the disjunctive graph: the first operation of the job 1 is o_1 and the fifth operation of the job 5 is o_{25}. For all 5 jobs, we can see that it is necessary to process for each machine the following operations:

- machine 1: 1, 2, 4, 9, 14, 17;
- machine 2: 5, 6, 7, 8, 10, 21;
- machine 3: 3, 11, 12, 13, 15, 18, 20, 25;
- machine 4: 16, 19, 22, 23, 24.

Figure 2 presents an optimal solution for this example, because the machine 3 defines $C_{\max}$ and it is not possible to reduce this time.

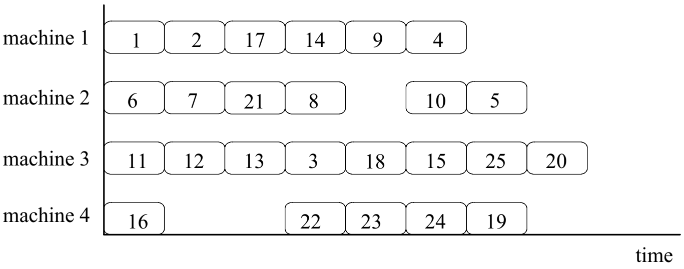


Figure 2. Optimal solution.

3 Proposed Methodology

In this work we adopted a method based on genetic algorithms. This technique's simplicity and easiness to model more complex problems combined with its easy integration with other optimization methods were factors that were taken into consideration in its choice. The algorithm proposed was conceived to solve the JSP, with or without recirculation.

3.1 Solution Representation

One of the features that differentiate conventional genetic algorithms is the fact that the algorithm does not deal directly with the problem's solutions, but with a solution representation, the chromosome. The algorithm manipulations are done over the representation and not directly over the solution (Goldberg 1989). The permutation code was chosen because it was adequate to permutation problems. In this kind of representation, the chromosome is a literal of the operations sequence on the machines. In the classical JSP case, the chromosome is composed by *m* sub-chromosomes, one for each machine, each one composed by *n* genes, one for each operation. In the JSP with recirculation case, the number of genes in each sub-chromosome is equal to the number of operations that the corresponding machine processes. The j^{th} gene of the sub-chromosome corresponds to the operation processed in j^{th} place in the corresponding machine. The allele identifies the operation's index in the disjunctive graph.

The Figure 3 illustrates the chromosome, which represents the solution presented in Figure 2.

machine 1	machine 2	machine 3	machine 4
1 2 17 14 9 4	6 7 21 8 10 5	11 12 13 3 18 15 25 20	16 22 23 24 19

Figure 3. Chromosome.

The manipulation of this type of chromosome through the conventional genetic operators can give rise to invalid solutions. Even the usage of especially designed operators for permutation problems can originate invalid solutions or solutions that are not active schedules. Due to this limitation, the information held in a chromosome can be seen as only one suggestion to the solution constructor algorithm. This option is justifiable since the JSP is hard to resolve, the search space is very extensive, and because it is well known that the optimum solution is an active schedule. In this way, the study of invalid solutions and semi-active schedules is considered a waste of time.

3.2 Solutions Decoding

The solutions are decoded by an algorithm, which is based on Giffler and Thompson's algorithm (Giffler and Thompson 1960) that creates active solutions. As advantages of this strategy we have pointed out minor dimension of solution space, which includes the optimum solution and the fact that it does not produce impossible or disinteresting solutions from the optimization point of view. On the other hand, since the dimensions between the representation space and the solution space are very different, this option can represent a problem because two chromosomes can represent the same solution.

The constructor algorithm has N stages and in each stage an operation is scheduled. To assist the algorithm's presentation, consider the following notation existing in stage *t*:

P_t - being the partial schedule of the $(t-1)$ scheduled operations;
S_t - being the set of operations schedulable at stage *t*;
σ_t - being the earliest time that operation o_k in S_t could be started;
ϕ_k - being the earliest time that operation o_k in S_t could be finished, that is $\phi_k = \sigma_k + p_k$;
M^* - Selected machine. Is the machine where $\phi^* = \min_{o_k \in S_k} \{\phi_k\}$;
S_t^* - Conflict set. Is formed by $o_j \in S_t$ so that will be processed in M^* and $\sigma_j < \phi^*$;
o_j^* - being the selected operation to be scheduled at stage *t*;
A_t - being the set of operations pointed by the chromosome at stage *t*.

The constructor algorithm of solutions is presented in a format, similar to the one used by French (1982) to present the Giffler and Thompson algorithm.

Algorithm 1: Constructor algorithm

```
Step 1 Let $t = 1$ with $P_1$ being null. $S_1$ will be the set of all operations with no
       predecessors; in other words those that are first in their job. $A_1$ will be
       the set formed by the first operations pointed in each sub-chromosomes.
Step 2 Find $\phi^* = \min_{o_k \in S_k} \{\phi_k\}$ and identify $M^*$. If there is a choice for $M^*$,
       choose arbitrarily. Form $S_t^*$.
Step 3 If $\exists\, o_j \in A_t \cap S_t^*$ then $o_j^* = o_j$, go to Step 5. Otherwise, go to Step 4.
Step 4 Select operation $o_j^*$ in $S_t^*$ randomly. Correct sub-chromosome of $M^*$,
       moving $o_j^*$ to the left pointed position.
Step 5 Move to next stage by
           (1) adding $o_j^*$ to $P_t$, so creating $P_{t+1}$;
           (2) deleting $o_j^*$ from $S_t^*$ and creating $S_{t+1}^*$ by adding to $S_t^*$ the
               operation that directly follows $o_j^*$ in its job (unless $o_j^*$ completes
               its job):
           (3) creating $A_{t+1}$ moving the pointer to the next right position in the
               $M^*$ sub-chromosome.
           (4) incrementing $t$ by 1.
Step 6 If there are any operations left unscheduled $(t < N)$, go to Step 2.
       Otherwise, stop.
```

3.3 Genetic Correction

To improve the method's efficiency we decided to use the genetic correction of the chromosome in order to represent the solution generated by the decoder. The correction of the chromosome is made inside of the Algorithm 1 in the Steps 3 and 4.

The main advantage of the chromosome correction is that it is possible to include the intrinsic characteristics of the solution, that is to say, it includes the relative order among operations of the same machine on the genetic code. At the stage of applying the genetic operators, the information about the solution's ordination is transmitted. It is intended that the choice of the set of conflicts is oriented with the help of the genetic algorithm. The indications of choice given by the good solutions must be propagated to the next generations.

The chromosome presented in Figure 3 just like the chromosome presented in Figure 4 gives origin to the solution of Figure 2. However, the sub-chromosomes of machine 4 are different. Throughout the iterations of the constructor algorithm this sub-chromosome would be successively corrected the way it is presented in Table 2.

machine 1	machine 2	machine 3	machine 4
1 2 17 14 9 4	6 7 21 8 10 5	11 12 13 3 18 15 25 20	24 16 19 22 23

Figure 4. Chromosome.

Table 2. Correction of sub-chromosome 4.

		Initial sub-chromosome	24 16 19 22 23
$S_4^* = \{16\}$	$A_4 = \{2,7,12,24\}$	$o_j^* = 16$	16 24 19 22 23
$S_{14}^* = \{22\}$	$A_{14} = \{9,10,18,24\}$	$o_j^* = 22$	16 22 24 19 23
$S_{17}^* = \{23\}$	$A_{17} = \{4,10,15,24\}$	$o_j^* = 23$	16 22 23 24 19
$S_{21}^* = \{19,24\}$	$A_{21} = \{5,25,24\}$	$o_j^* = 24$	16 22 23 24 19
$S_{24}^* = \{19\}$	$A_{24} = \{19,20\}$	$o_j^* = 19$	16 22 23 24 19

3.4 Genetic Operators

In this algorithm three genetic operators are used, JOX, LS-swap and PSM. The operators JOX and PSM were chosen based in the results obtained in the previous computation experiments (Oliveira 2001). The binary operator JOX (Job-based Order Crossover) was presented by Ono et al. (1996). The authors point out as an advantage of this operator the preservation of the progenitor's characteristics on the descendants. JOX can preserve the order of each job on all machines properly between parents and their offspring.

The LS-swap operator is a unary operator (implicates only one solution) and its objective is to include specific knowledge of the problem into the genetic algorithm under the shape of local search of solutions. This operator exchanges two operations according to the neighbourhood structures that exist on the JSP literature, based on blocks of the critical path. This operator promotes a controlled reproduction that aims to reach local minima. This process is different from the procedures that exist on memetic algorithms, because the obtainment of the local optimum is not guaranteed in one iteration / generation only. The obtainment of the local minimum is fruit of an evolutionary process throughout iterations. This strategy's objective is to avoid the premature convergence and to maintain some genetic diversity on the population.

For neighbourhood structure the model proposed by Nowicki and Smutnicki (1996) was chosen. The fact that this neighbourhood scheme is simpler was what lead to it's choice.

Algorithm 2: LS-swap(chromosome, blocks)

```
begin
    select one block;
    select the extremity of the block; //beginning or end
    swap the two extreme operations;
end
```

Figure 5 presents a solution and point out the operations that form the critical path of the schedule, and shows the existence of 3 critical blocks of operations.

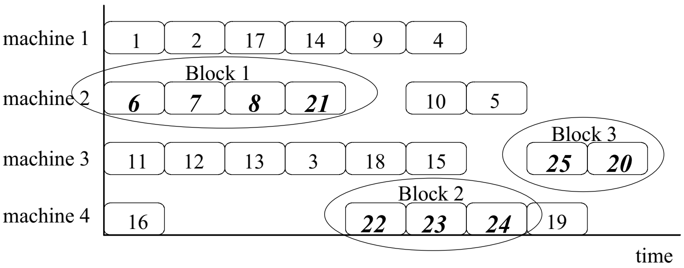


Figure 5. Critical path and blocks.

Figure 6 shows the chromosome that represents the solution for Figure 5. All three blocks of the solution are represented in ***bold italic***. According to the neighbourhood scheme proposed by Nowicki and Smutnicki (1996), the swap of the operations 8 and 21 in Block 1 is allowed and in this case leads to the optimal solution presented in Figure 2. However, we point out that in JSP with recirculation, in block 2 it is not possible to swap the operations because all operations of the block belong to the same job.

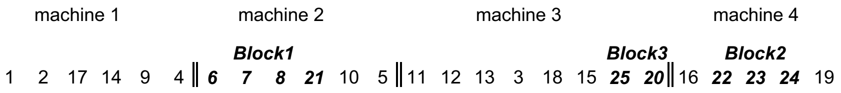


Figure 6. Chromosome and the critical blocks.

The mutation operator used is the PSM (Permutation Swap Mutation). This operator exchanges the alleles of two genes selected randomly belonging to the same sub-chromosome, randomly selected itself.

3.5 The Genetic Algorithm

The genetic algorithm has a very simple structure and can be represented in the followed simplified way:

```
Algorithm 3: GALS_swap
begin
P ← GenerateInitialPopulation()
Evaluate(P)
while termination conditions not meet do
    P' ← Recombine(P)  //60% JOX + 40% LS_swap
    P'' ← Mutate(P')  //PSM
    Evaluate(P'')
    P ← Select(P ∪P'')
end while
```

4 Computational Experiments

Computational experiences were carried out with five instances of the JSP with recirculation generated randomly. Instances Jr11, Jr12, Jr13, Jr14 and Jr15 are constituted by 13 jobs and 35 operations per job, which adds up to a total of 455 operations and 11 machines. The operations of one of the job were distributed at random by a group of machines according to the percentages defined on Table 3. All operations o_i have $p_i = 1$.

Table 3. Operations distribution (%)

	Machines										
Jobs	1	2	3	4	5	6	7	8	9	10	11
1	50	30	20								
2	25	25	25	25							
3	25	25	25	25							
4		25	25	25	25						
5		20	20	20	20	20					
6			20	20	20	20	20				
7				20	20	20	20	20			
8					20	20	20	20	20		
9						20	20	20	20	20	
10							25	25	25	25	
11								25	25	25	25
12								25	25	25	25
13									20	30	50

On Table 4 the obtained computational results are presented. The algorithm was implemented in Visual C++ 6.0 and the tests were run on a computer AMD Athlon(TM)XP 1400+, with 512 MB of RAM, on the MS Windows 2000.

Table 4. Results for 5 instances, randomly generated

		GALS-swap						
		(1)	(2)	(*)	EDD	LRPT	SRPT	FCFS
Instance		mean5	mean5					
(Size)	LB(M)	CPU	CPU					
Jr11	52	57.8	57.8	57	66	64	81	66
(455)		73	441					
Jr12	53	60.4	59.4	58	77	65	79	67
(455)		73	441					
Jr13	60	62	61.4	61	77	63	82	65
(455)		73	441					
Jr14	53	59.4	59.2	59	69	62	78	66
(455)		73	441					
Jr15	49	58	57.4	57	67	62	73	63
(455)		73	441					

The second column presents lower bound calculated in a very elementary way. LB(M) means the biggest operations number (duration) that a given machine has to process. The equivalent of this lower bound calculated in the instance FT10 is 631 time units, whereas the value of the optimal solution is 930.

In configuration (1) the result is the average of five runs with a population of 100 individuals and 500 iterations. In configuration (2) the result is the average of five runs with a population of 300 individuals and 1000 iterations. In column (*) the best solution found is presented.

Columns EDD, LRPT, SRPT e FCFS represent the value of active schedules built based on dispatch priority rules. EDD means Earliest Due Date, which in this case means shortest index job. LRPT means Longest Remain Processing Time, SRPT means Shortest Remain Processing Time and FCFS means First Come First Served (French 1982).

Table 5 presents the results of computational experiences carried out with test instances of the classical JSP (Beasley 1990) chosen by Vaessens et al. (1996). The algorithm's configurations are the same used for the JSP with recirculation, whose results are presented in Table 4.

The references in the columns means:

- GMR	(Gonçalves 2002);
- AF	(Aydin and Fogarty 2002);
- ABR	(Aiex et al. 2003);
- BHLR	(Binato et al. 2001);
- GB	(Gonçalves and Beirão 1999);
- Kol	(Kolonko 1999);
- NS	(Nowicki and Smutnicki 1996).

Table 5. Results for some of the hard instances

		Oliveira *GALS-swap*			GMR (2002)	AF (2002)	ABR (2001)	BHLR (2002)	GB (1999)	Kol (1999)	NS (1996)
		(1)	(2)	(*)							
Instance	UB	mean5	mean5	best	n.a.	mean	best	best	n.a.	mean5	mean1
(Size)	LB	*CPU*	*CPU*		*CPU*	*CPU*	*CPU*	*CPU*	*CPU*	*CPU*	*CPU*
FT10	930	939	931.6	930	930		930	938	936		930
(100)	930	*6*	*37*		*292*		*1.01E4*	*2.61E5*	*n.a.*		*30*
La2	655	655	655	655	655		655	655	666		655
(50)	655	*3*	*17*		*51*		*1.10E1*	*1.40E2*	*n.a.*		*8*
La19	842	847.4	842	842	842		842	842	857		842
(100)	842	*6*	*36*		*235*		*3.03E2*	*3.13E4*	*n.a.*		*60*
La21	1046	1082	1062.6	1058	1046	1048.4	1057	1091	1047	1051	1055
(150)	1046	*11*	*68*		*602*	*838*	*3.51E5*	*3.26E5*	*n.a.*	*549*	*21*
La24	935	968	959.8	955	953	939.6	954	978	955	940.4	948
(150)	935	*11*	*67*		*578*	*570*	*4.08E5*	*6.46E4*	*n.a.*	*570*	*184*
La25	977	1002.6	994.6	988	986	977.8	984	1028	1004	979	988
(150)	977	*11*	*67*		*609*	*1035*	*1.05E5*	*6.46E4*	*n.a.*	*644*	*155*
La27	1235	1302.4	1270.2	1263	1256	1245.4	1269	1320	1260	1244.8	1259
(200)	1235	*18*	*106*		*1251*	*982*	*2.89E5*	*1.10E5*	*n.a.*	*3651*	*66*
La29	1153	1256	1222.8	1212	1196	1182.6	1203	1293	1190	1169.2	1164
(200)	1142	*18*	*106*		*1350*	*1148*	*3.09E5*	*1.12E5*	*n.a.*	*4494*	*493*
La36	1268	1313.2	1311.2	1295	1279		1287	1334	1305	1269.8	1275
(225)	1268	*19*	*112*		*1826*		*2.83E5*	*1.15E5*	*n.a.*	*4655*	*623*
La37	1397	1458	1447.8	1442	1408		1410	1457	1441	1412.8	1422
(225)	1397	*19*	*113*		*1860*		*2.50E5*	*1.15E5*	*n.a.*	*4144*	*443*
La38	1196	1252.8	1242.2	1228	1219	1214.6	1218	1267	1248	1202.4	1209
(225)	1196	*19*	*113*		*1859*	*1143*	*6.57E5*	*1.19E5*	*n.a.*	*5049*	*165*
La39	1233	1271.2	1250.4	1249	1246		1248	1290	1264		1235
(225)	1233	*19*	*112*		*1869*		*3.54E5*	*1.15E5*	*n.a.*		*325*
La40	1222	1257.2	1250.2	1234	1241	1229.2	1244	1259	1252	1228.6	1234
(225)	1222	*19*	*113*		*2185*	*1895*	*2.08E5*	*1.23E5*	*n.a.*	*4544*	*322*

Table 5 shows an interesting balance between effectiveness and efficiency for the GALS-swap. Although the computation times cannot be directly comparable because they were carried out in different machines, it can be verified that the results obtained in this genetic algorithm do not go beyond the average of 2.4% of the Lower Bound, in a very interesting computation time.

In configuration (1) we get an average error of 3.47% in 172 seconds. In configuration (2) we get an average error of 2.37% in 1031 seconds. Considering the goal stated by Vaessens et al. (1996) (to achieve an average error of less than 2% within 1000 seconds total computation time, for this set of instances), we realise that the obtained results are very close from the goal. Further more the CPU time can get better if the algorithm is stopped, for example, after 100 iterations without any improvement.

For the best obtained values, the GMR's algorithm presents results that on average are 0.6% better than GALS-swap algorithm. According to Vassens et al. (1996), the GALS-swap algorithm uses a computation time that is 2% of the computation time of GMR's algorithm, because for the best obtained result it corresponds a computation time that is the sum of all runs; for the average value it corresponds the computation time of a single run.

5 Conclusions

This paper presents a genetic algorithm to solve either the JSP with recirculation or the classical JSP. The schedules are built using information given by the genetic algorithm to order the operations.

The algorithm incorporates a local research procedure with the shape of a genetic operator. This operator allows the inclusion specific knowledge of the problem and makes the algorithm very efficient.

The approach is tested on a set of standard JSP instances taken from the literature and considered the more difficult ones. The computational results show that the algorithm produced good solutions on all instances tested. Overall, the algorithm produced solutions with an average relative deviation of 2.4% from the best-known solution. This performance allows to foresee a similar performance with the JSP with recirculation.

To increase the effectiveness of our algorithm we can use a population of larger dimension, or to include an auxiliary procedure of local search, or to hybridize with other strategies such as in the tabu search.

References

Aiex, R.M., Binato, S., and Resende, M.G.C. (2003), “Parallel GRASP with path-relinking for job shop scheduling,” *Parallel Computing*, vol. 29, pp. 393-430.

Aydin, E., and Fogarty, T.C. (2002), “Modular simulated annealing for job shop scheduling running on distributed resource machine (DRM),” *Technical*

Report, South Bank University, London, (Downloadable from website http://www.dcs.napier.ac.uk/~benp/dream/dreampaper13.pdf).

Beasley, J.E. (1990), “OR-Library: distributing test problems by electronic mail,” *Journal of the Operational Research Society*, vol. 41, pp. 1069-1072. (website http://mscmga.ms.ic.ac.uk/info.html).

Binato, S., Hery, W.J., Loewenstern, D.M., and Resende, M.G.C. (2001), “A GRASP for job shop scheduling, in: C.C. Ribeiro, P. Hansen, (Eds.), *Essays and Surveys in Metaheuristics*, Kluwer Academic Publishers, Dordrecht, pp. 58-79.

Brucker, P. and Knust, S. (2003), *Complexity results for scheduling problems*, Department of Mathematics/Computer Science, University of Osnabrück, (Downloadable from website http://www.mathematik.uni-osnabrueck.de/research/OR/class/).

French, S. (1982), *Sequencing and Scheduling - An Introduction to the Mathematics of the Job-Shop*, Ellis Horwood Limited, Chicester.

Garey, M., Johnson, D.S., and Sethi, R. (1976), “The complexity of flowshop and jobshop scheduling,” *Mathematics of Operation Research*, vol. 1, pp. 117-129.

Giffler, B. and Thompson, G.L. (1960), “Algorithms for solving production scheduling problems,” *Operations Research*, vol. 8, pp. 487-503.

Goldberg, D.E. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

Gonçalves, J.F. and Beirão, N.C. (1999), “Um algoritmo genético baseado em chaves aleatórias para sequenciamento de operações,” *Revista Associação Portuguesa de Desenvolvimento e Investigação Operacional*, vol. 19, pp. 123-137.

Gonçalves, J.F., Mendes, J.J.M., and Resende, M.G.C. (2002), “A hybrid genetic algorithm for the job shop scheduling problem,” *AT&T Labs Research Technical Report TD-5EAL6J*, Florham Park, NJ, (Downloadable from website http://www.research.att.com/~mgcr/doc/hgajss.pdf).

Jain, A.S. and Meeran, S. (1999), “A state-of-the-art review of job-shop scheduling techniques,” *European Journal of Operations Research*, vol. 113, pp. 390-434.

Kolonko, M. (1999), “Some new results on simulated annealing applied to the job shop scheduling problem,” *European Journal of Operational Research*, vol. 113, pp. 123-136.

Lenstra, J.K. and Rinnooy Kan, A.H.G. (1979), “Computational complexity of discrete optimization problems,” *Annals of Discrete Mathematics*, vol. 4, pp. 121-140.

Nowicki, E. and Smutnicki, C. (1996), “A fast taboo search algorithm for the job-shop problem,” *Management Science*, vol. 42, pp. 797-813.

Oliveira, J.A. (2001), *Aplicação de Modelos e Algoritmos de Investigação Operacional ao Planeamento de Operações em Armazéns*, Ph.D. Thesis, Universidade do Minho, Braga.

Ono, I., Yamamura, M., and Kobayashi, S. (1996), “A genetic algorithm for job-shop scheduling problems using job-based order crossover,” *Proceedings of ICE'96*, pp. 547-552.

Roy, B. and Sussmann, B. (1964), “Les Problemes D’Ordonnancement Avec

Vaessens, R.J.M., Aarts, E.H.L., and Lenstra, J.K. (1996), “Job Shop Scheduling by local search,” *INFORMS Journal on Computing*, vol. 8, pp. 302-317. (Downloadable from web site http://joc.pubs.informs.org/BackIssues/Vol008/Vol008No03Paper09.pdf).

A Multiobjective Metaheuristic for Spatial-based Redistricting

Bong Chin Wei

Faculty of Computer Science and Information Technology,
Universiti Malaysia Sarawak,
94300 Kota Samarahan, Sarawak, Malaysia
cwbong@fit.unimas.my

Abstract. This study has developed and evaluated a multiobjective metaheuristic for redistricting to draw territory lines for geographical zones for the purpose of space control. The proposed multiobjective metaheuristic is briefly explained to shows its components and functionality. The redistricting problem definition is discussed, followed by the metaheuristic and the multiobjective decision rules. Then, an experiment is conducted in Geographic Information System (GIS) to shows its performances especially in term of its quality of result. The focus of the experiment is on the performance analysis of the coverage of the approximately non-dominated solution set and the number of objectives defined. The result of the experiment has demonstrated an improvement.

Keywords: Soft Computing for Modeling and Optimization, Industrial Applications

1 Introduction

Redistricting is defined as drawing lines for the purpose to partition geographical zones into territories subject to some side constraints [4,5]. This paper reports redistricting decision-support with a multiobjective metaheuristic. Redistricting is extremely important as it determines the management of the space control of a particular region [1]. The process is relevant in election, school, business, law enforcement and even forest planning. Redistricting is a real-life optimization problem that requires taking into account multiple points of views corresponding to multiple objectives. Various criteria exist to evaluate the quality of the given solution and thus, there is an objective (minimization or maximization) attached to each of the criteria. However, there is often a variety of competing objectives in real word situations. “Perfect solution for achieving any one of competing objectives may be impossible” [9]. Although there were various studies in the development of multicriteria spatial decision-making tools and methodologies from 1998 to 2002, research exploration in the specific spatial multiple objective redistricting is rare [8]. Existing studies do not provide technical feasibility, theoretical extension and refinement for spatial multiple objective optimization. Conversely, the “No free lunch” theorem for optimization clearly indicates that no method may outperform all the other methods on all problems [11].

The rest of this paper is organized as follow. In Section 2, the multiobjective metaheuristic for redistricting is described. This is followed by implementation and application in Section 3 and by conclusions in Section 4.

2 The Multiobjective Metaheuristic

This study proposes a design of redistricting with multiobjective problem definition and measurement (or approximation) of non-dominated solutions. A metaheuristic composes of the ideas from tabu search, scatter search and path relinking is proposed to improve the search process. Symbioses of the existing metaheuristics help support spatial data and process with intensification and diversification process. The proposed method aims to identify wider range of alternatives and to consider the relationship between alternatives. The following describes the components and functionalities of the framework from the basic understanding and definition, initialization, neighboring move, subset generation and combination, multiobjective decision rules and measurement used.

2.1 General Redistricting Problem Definition

Several heuristic have been proposed for the districting problem to combine indivisible basic units into feasible districts. This formulation can be serving as a guide for a heuristic. Therefore, this research chooses to formulates the problem as an assignment problem that modelled as a capacitate *m*-median problem as follows:

$$\min \sum_{i\in I}\sum_{j\in J} c_{ij}x_{ij} \tag{1}$$

In term of multiobjective definition, this study includes a set of *n* spatial decision variables, three objective functions, and a set of *m* constraints for the general redistricting problem definition. The objective functions are functions of the decision variables subject to the constraints. The optimization goal is defined as following:

$f(x) = \min(y)$

where $y = (y_1, y_2, y_3) \in Y$, *y* is the objective vector, ,*Y* is the objective space

$x = (x_1, x_2, \ldots, x_n) \in X$, *x* is the decision vector, *X* is the decision space

I = the set of all basic units. For each unit, all capacity data and geographical data are available.

J = the set of basic units used as potential ‘‘seeds’’.

m = the number of zones to be created is given

p_i = the capacity of unit *i*

[*a, b*] = interval of the capacity of any zone must lie.

The decision variables considered is to let x_{ij} be a binary variable equal to 1 if unit *i* is assigned to seed *j*.

$$x_{ij} = 0 \qquad or \qquad 1 \qquad (i \in I, j \in J) \tag{2}$$

The constraints include:
(1) Each basic unit is assigned to one district,

$$\sum_{i \in I} x_{ij} = 1, (i \in I) \tag{3}$$

(2) The number of districts is equal to *m*

$$\sum_{j \in J} x_{jj} = m \tag{4}$$

(3) No basic unit can be assigned to an unselected seed

$$x_{ij} \le x_{jj}, (i \in I, j \in J) \tag{5}$$

(4) Resources capacity is taken into account

$$a \le \sum_{i \in I} p_i x_{ij} \le b, (j \in J) \tag{6}$$

This study then refers an objective function vector as a *point* z, so where $z^x = [z_1^{\ x}, ..., z_j^{\ x}] = f(x)$ with $j = 1,...,J$. Let us also define dominance and superiority [10]:

The point z_1 *dominates* the point z_2 if and only if $z_1 > z_2$. The point z_1 is *dominated* by the point z_2, if the point z_2 dominates the point z_1. If any other points do not dominate a point, it is called a *non-dominated* point.

The set of all non-dominated points is referred as the non-dominated set. An efficient solution for this study should be Pareto optimal, and the solutions are uniformly sampled from the Pareto-optimal set. Also, this study uses weight vector (λ) from the λ-vector space Λ for each of objective functions that defined as

$$\Lambda = \{\lambda \in \Re^j \mid \lambda^k \in [0;1] \ni \sum_{j=1}^{J} \lambda^j = 1\} \tag{7}$$

Range equalization factors are used to equalize the ranges of the objectives, and calculated as

$$\pi_j = \frac{1}{R_j} \qquad , j = 1,..., J \tag{8}$$

where R_j is the (approximate) range of objective *j* given a set of points. Objective function multiplied by range equalization factors is called normalized objective function value.

2.2 Seed Solution Initiator

A wide exploration of the solution space is important to navigate the algorithm into various regions of the search domain. Therefore, the first step in the proposed approach is to create an initial solution with diversification purpose to encourage the search process to examine random regions of the solution space. Given a set *I* of basic units and attributes in a layer form, this study generates a set of initial solution with an algorithm of seed solution generator with random diversification. Firstly, a seed unit is selected randomly to initialize a zone. Then, this zone is extended gradually by adjoining to one of its adjacent units. The zone is complete whenever no adjacent units are available or when its capacity attains the balance among the desired number of districts. When there is any basic unit that has not been assigned with a zone number, they will be merged with least populated zone. If all zones in the final zoning plan are continuous, it becomes the initial seed solution. In other words, at the end of this process, the initial solution x_0 is a zoning plan made up of *m* continuous zones, some of which may be infeasible with respect to some of the objectives defined.

2.3 Neighbouring Move and Generated Subset Combination

After this study has obtained the initial solution in a zoning plan, the proposed algorithm conducts a neighboring tabu move strategy to prepare for generating an intensified solution to return to attractive regions of the solution space to search them more thoroughly. In this case, this study adopts the move strategies used by [5] with two neighborhoods move. The first move is made up of all solutions reachable from *x* by moving a basic unit from its current zone to a neighbor zone without creating a non-contiguous solution. Such a move is said to be of Type I. The second neighborhood move is made up of all solutions that can be reached from x by swapping two border units between their respective zones, again without creating discontinuities. Such a move is said to be of Type II. On the other hand, to help prevent cycling, whenever a Type I or Type II move is performed, any move that puts the basic unit back into its original zone is declared as tabu. A ϕ value in the scalarizing function in the multiobjective decision-making is a tabu daemon that helps enhance the proposed hybrid metaheuristic. The tabu daemon overrides the tabu status of a move when it yields the best solution obtained up till then [2]. Usually tabu daemon is also referred as “aspiration criterion” [7]. In other words, it is satisfied if the function value reached after the move is better than the best found previously [3]. In that case, ϕ is penalty term related to problem size. Therefore, $\phi = 1/\sqrt{p}$, where *p* is the number of the basic unit used. Choosing $\sqrt{p}$ rather than *p* for example is common for the idea to use a multiplier that reflects the problem size. It was observed empirically that *p* overemphasizes problem size whereas $\sqrt{p}$ produces a smoother search [5].

By moving until hitting a feasible boundary, where normally a heuristic method would stop, this study extends and evaluates the neighborhood definition for selecting modified moves, to permit crossing of the boundary. These strategies are used respectively to focus the search into more promising regions, modifying choice rules to stimulate combinations of good moves and improved solutions. It leads the search to unexplored regions of the solution space and generates a neighboring subset. This study combines the subset with spatial adjoining or merging process. In other words, the neighboring move provides a pool of basic units for generating neighboring subsets to construct solutions by combining various elements with the aim that the solution based on the combined elements will exploit features not contained separately in the original elements. The neighboring subsets contain dynamic number of basic units depending on the adjacent units and this helps to reduce the computation time in the procedure. Therefore, the neighboring subset generator and combination work with the underlying concept of scatter search but have intimate association with tabu search metaheuristic. The solution combination method transforms the given subset of solutions produced by the dynamic neighboring subset into a combined solution.

2.4 Multiobjective Decision Rules and Measurement

This study determines the weight vector (λ) for each of the objective function vectors or points. The neighboring move will first ensure optimization towards the non-dominated frontier. Then, this study will set and modify the weights, which are obtained from the decision-maker, so that the point moves away from the other points, ideally having the points equidistantly spread over the frontier. Therefore, each element in the weight vector is set according to the proximity of other points for that objective. However, this study only compares a point with the points of the current solution to which it is non-dominated. The closer another point is, the more it should influence the weight vector. The closeness is measured by a distance function (*d*) based on some metric in the objective function space and the range equalization weights. The influence is given by a decreasing, positive value of proximity function (*g*) on the distance. In practice, the proximity function *g(d)=1/d* has shown to work well, as well as the Manhattan distance norm as in equation (8). The distance norm, π used on the objectives is then scaled by the range equalization factors.

$$d(z_i^k, z_j^k, \pi) = \sum \pi^k \mid z_i^k - z_j^k \mid \tag{9}$$

The use of a multiobjective acceptance rule in a quality measurement is crucial in approximating the non-dominated solution for the multiobjective redistricting problem. Therefore, this study uses the quality dimension in tabu search that usually refers to the ability to differentiate the merit of solutions visited during the search. In this context, this study uses memory to identify elements that are common to good solutions or to paths that lead to such solutions. Operationally, quality becomes a foundation for incentive-based learning, where inducements are provided to reinforce actions that lead to good solutions. The flexibility of the memory structures allows the search to be guided in a multiobjective environment, where the goodness of a particular search direction may be determined by more than one functions. This study concentrates on the quality counter with achievement scalarizing function, which takes into account the weight vector, reference set of the optimal solution and each objective function scaling. The advantage of scalarizing function is the possibility of forcing particular solutions to explore the desired regions of non-dominated set. It allows for problem specific heuristics for construction of initiate solution and/or local improvement.

Then, for the purpose for performance comparison, this study uses Achievement Tchebycheff Scalarizing Function (ATSF) to qualify the generated solution with the reference point, at the objective functions f(x) or z where z^0 is a reference point, $\Lambda = [\lambda_1, \ldots, \lambda_j]$ is weight vector, and ϕ is tabu daemon mentioned in previous section. ATSF is defined as (10) and the use of ATSF is good at locating non-supported non-dominated points [10].

$$S(z, z^0, \lambda, \phi) = \max_j \{\lambda_j (z_j{}^0 - z_j)\} + \phi \sum_{j=1}^{j} \lambda_j (z_j{}^0 - z_j) \tag{10}$$

Besides, this study chooses to use interactive procedures or modes for guiding the search by using a reference point from [6]. The preferences would first obtain through a dialogue prompt from decision-maker (DM) and then aggregated with range equalization factors and a γ-parameter range from 0 to 1 to define the intensification of the search in the reference direction on behalf the diversification of the current solution and therefore in the resulting approximation. The aggregation of a reference weight vector, r with range equalization factors and a γ-parameter is given in equation below:

$$\lambda^k = \gamma d^k + (1 - \gamma)\, \lambda^k \tag{11}$$

where $d^k = \pi^k r^k$

The solutions set obtained from the approximation process are handled in a reference set. The proposed method stores the constantly updated set of potentially pareto-optimal solutions in a Reference Set (*RS*). *RS* is empty at the beginning of the method. The scheme continuously updates it whenever a new solution is generated. Updating the set of potentially Pareto-optimal solutions with solution x consists of adding z to *RS* if no point in *RS* dominates z, and removing from *RS* if all points dominated by z. New solution y obtained from x should be used to update *RS* only if it is not dominated by x. New potentially pareto-optimal solutions could be added to *RS* only if they differ enough from all solutions contained already in this set. This study uses a threshold defining the minimum Euclidean distance in the space of normalized objectives between solutions in *RS*. A new potentially pareto-optimal solution is neglected if it is closer to at least one solution in *RS* than the threshold. The solutions added to *RS* in early iterations have a good chance to be removed from this set in further iterations. Whenever a new solution is created, the zoning plan becomes the member of the reference set memory. The size of the memory is kept constant that its worst elements are regularly replaced by better ones.

3 Experiment

An experiment was conducted in Visual Basic Application (VBA) embedded in ArcGIS and run on a Pentium IV 2.4 GHz PC with 256MB RAM. ArcGIS, a high-end GIS software, is flagship product of Environmental Software Research Institute (ESRI), which has the capabilities of automation, modification, management, analysis and display of geographical information. The input data to the multiobjective redistricting problem was stored in the form of map layers called Shapefiles format, which were handled and visualized using the ArcGIS. Each map layer contained a set of objects that were considered as elements of alternative solutions. The map layers were first processed to define the redistricting objectives as mentioned earlier and the underlying attributes of the objectives contained in the geographical space. Some of the basic operations used included operations on spatial relationship of connectivity, continuity, proximity and the overlay methods. As it was not possible to use the standard operations alone to generate the solutions, the proposed multiobjective metaheuristic was specifically designed, coded and aggregated in the VBA code to tackle multiobjective redistricting problem. The algorithm was coded and tested on a political districting. Three objectives are considered in this model. Each of the following functions corresponds to one of the goals.

Goal 1: The objective function f_1 corresponding to the goal 1 measures the average deviation of the population. Indeed, population equality, $P_{\mathrm{j}}(x)$ is the population of district j, the average district population of each district is $\overline{P} = \sum_{j\in J} P_j(x)/m$. The population of each district lies within some interval [*a*, *b*] = $[(1-\beta)\overline{P}, (1+\beta)\overline{P}]$ where $0 \le \beta < 1$. The objective is formulated in the following way:

$$f_1 = \frac{\sum_{j\in J} \max\{P_j(x) - (1+\beta)\overline{P}, (1-\beta)\overline{P} - P_j(x), 0\}}{\overline{P}} \tag{12}$$

Goal 2: The objective function f_2 corresponding to the goal 2 measures compactness by measuring the total length of all boundary lengths between zones, excluding the outside boundary of the territory:

$$f_2 = \sum_{j\in J} (1 - \frac{2\pi\sqrt{A_j(x)/\pi}}{R_j(x)}) \Big/ m \tag{13}$$

where $R_j(x)$ and $A_j(x)$ are the perimeter and area of j in the solution x.

Goal 3: The objective function f_3 corresponding to the goal 3 measures a socio-economic homogeneity, *S.* the reasonable objective is to minimize the sum over all zones *j*, of the standard deviation $S_{\mathrm{j}}(x)$ of by the average income of each basic unit in the zone.

$$f_3 = \sum_{j\in J} S_j(x)/\overline{S} \tag{14}$$

Since it is unrealistic to test all combinations of candidate values for every aspect of the proposed multiobjective metaheuristic for redistricting in this paper, the focus of the experiment is to give an understanding on the behavior and performance of the multiobjective decision rules and measurement of the proposed multiobjective metaheuristic. The study considers 3 zones created for each zoning plan and uses 55 basic units for the input of the model (refer Fig. 1).

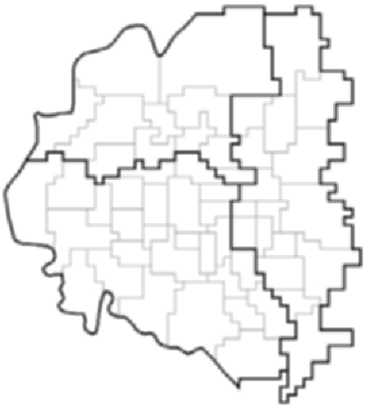


Fig. 1. Initial zoning plans and the basic units used.

In the multiple objective case, a set of potentially efficient solutions may be compared with randomly generated solution, another solution set obtained by different methods or some reference set of solutions such as the set of efficient solutions of the best approximation known so far. The quality metrics described in this paper concern as the measurement of the set of efficient solutions for the best approximation. The method is used for quality assessment and it is a geometric quality measures. It is to measure "distance" between the actual non-dominated set in the RS on the approximation of the non-dominated set. Therefore, this study tends to use assessment method from [10] and [11] for the purpose above to measure the average value of each point in RS. The measurement methods are as following

a) Distance measure based on a RS

$$Dist1 = \frac{1}{|RS|}\sum_{r\in RS}\min_{z\in A}\{s_\infty(z,r,\Lambda)\} \tag{15}$$

where $s_\infty(z,r,\Lambda) = \max_{j=1,\dots,J}\{0,\lambda_j(z^j - r^j)\}$

b) Consider worst-case distance from a reference point to its closet neighbour in A

$$Dist2 = \max_{r\in RS}\min_{z\in A}\{s_\infty(z,r,\Lambda)\} \tag{16}$$

They are used together for measure of uniformity of the quality of measurement

$$\text{Uniformity} = Dist2/Dist1 \tag{17}$$

The first metric *Dist1* gives information about the average distance from $y \in \Re$ to the closet solution in RS while the second solution give the worst case. The lower the values the better the set approximates the solution. Moreover, the lower the ratio *Dist*2/*Dist*1, the more uniform the distribution of the solution from set RS over the set of non-dominated solution.

3.1 Analysis of the Coverage of Approximation of Non-Dominated set

The coverage of the approximation of the non-dominated set is affected by several factors. These factors include the dominancy comparison, the tuning of the p-value in the achievement scalarizing function and the γ-parameter in the decision-makers' preferences. Therefore, the following subsections aims to analyze these factors in producing the solutions in the RS.

3.1.1 Analysis of the Distance Measurement for Dominancy Comparison

In the analysis of distance measurement for dominancy comparison, two different dominancy comparison methods have been conducted for investigating its effects in the proposed framework. The different comparison methods selected are concerning the definition of the weak dominance and the strong dominance. Both of these dominancy comparison methods are implemented in a rule set. The weak dominance is defined in section 2.1. For implementing the weak dominance, the rule set is given in Fig. 2. The rule set for strong dominance is given in Fig. 3.

If $f_1(y) <= f_1(\mathrm{x})$ AND $f_2(y) <= f_2(\mathrm{x})$ AND $f_3(y) <= f_3(\mathrm{x})$ AND _
($f_1(y) < f_1(\mathrm{x})$ OR $f_2(y) < f_2(\mathrm{x})$ OR $f_3(y) < f_3(\mathrm{x})$) Then

Fig. 2. The rule set for the weak dominancy comparison

If $f_1(y) < f_1(\mathrm{x})$ AND $f_2(y) < f_2(\mathrm{x})$ AND $f_3(y) < f_3(\mathrm{x})$ Then

Fig. 3. The rule set for the strong dominancy comparison

The result of the experiment of the two different contexts is presented for their respective *Dist1* values. Different number of seed solutions is used to generate the result for the experiments in order to verify the influence of the different dominancy comparison methods. A graph has been plotted in Fig. 4 to check the best point and patterns of changes when number of seeds solution increased. From the analysis of the result, at the beginning, the *Dist1* values of the strong dominance are higher than weak dominance. Thus, the distances measured for the reference set are relatively low in the strong dominance compared to the weak dominance. After the number of seed solution reach 5 or 6, the strong dominance gives lower value compared to the weak dominance. In other word, it is found that the best result is produced for strong dominance when the number of seed equal to 6 while for weak dominance is 5. For the two different types of dominancy measurement, strong dominance in the distance measurement gives a better result with higher number of seed solutions. Since it is uncommon to use less seeds in solution generation, strong dominance gives a more preferable result.

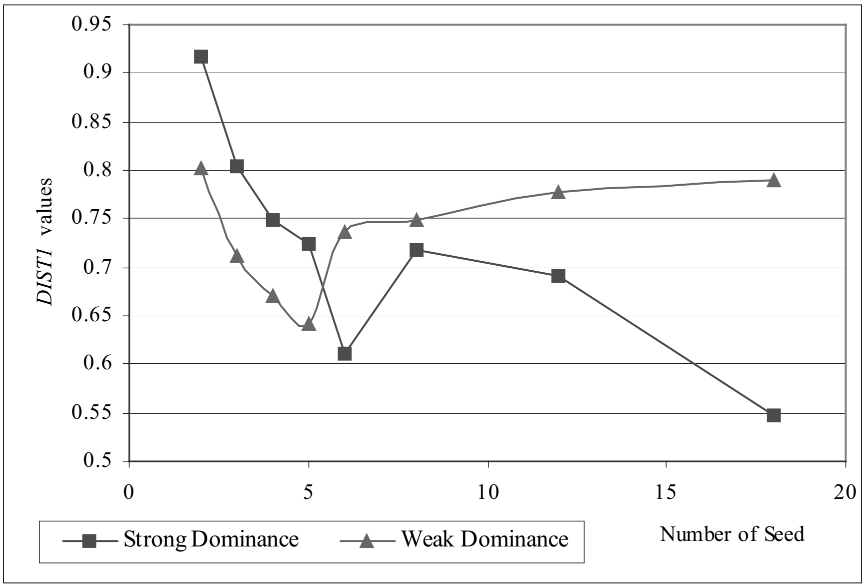


Fig. 4. *DIST1* for strong and weak dominance with different number of seed solutions

3.1.2 The ϕ-value

The analysis on the ϕ-value in the Achievement Tchebycheff Scalarizing Function (ATSF) is surveyed. Firstly, the value is tested with constant value, then followed by value depend on problem sizes and number of desired districts formed. With a constant value, if the value is small enough and is depending on the number of basic units and size of the desired districts form, the result is obviously better. When the value considers the number of basic units, the result gives a better result of 3 % compared to the result without the consideration of the number of basic unit in the problem size. On the other hand, when the value equals to 1/sqr(m), the ATSF gives the distance measure, *DIST1* which is similar to the result given by the Weighted Tchebycheff Scalarizing Function (WSF) defined in (18). In short, the -value is useful in controlling the result in the method. The actual setting of the parameter allows an overall scaling of the penalty term that related to the problem size. The idea is to use a multiplier that reflects problems size. It was observed empirically that m overemphasizes problem size whereas sqr(m) produces a smoother search. As shown in the Fig. 5, ATSF with the ϕ-value that considers the problem size and the number of desired output give the best result.

$$S(z, z^0, \Lambda) = \max_j \{\lambda_j (z_j^{\ 0} - z_j)\} \tag{18}$$

3.1.3 The γ-parameter

The γ-parameter is important in controlling the intensification of the search in the reference direction on behalf the diversification of the current solution and therefore in the resulting approximation. As it is aggregated with the weight vector, it will influence the result of the approximation process. Therefore, this research tends to analyse the different value of the γ-parameter to the effects of the overall result. This study attempts to use the gradually increase values range from 0.005, 0.025, 0.05, 0.25, 0.5, and 1.5 in order to examine the result. According to the result as shown in Fig. 6, the minimum *DIST1* value is found when the γ-parameter equals 0.05. In other words, this is the best value of the parameter in producing the shortest distance in the measurement of the RS. When the γ-parameter value increases gradually, the result is worst and not satisfying because its *DIST1* values increase dramatically.

3.2 Analysis of the Number of Objectives Defined

The survey on the number of objectives defined in the proposed method is analysed. Number of objectives has been investigated in order to study the complexity and the consistency of the proposed framework when the number of objectives defined increases. Three cases on three, four and five objectives have been conducted for this purpose to compare the distance measurement in the reference set, worst-case analysis, and the uniformity. The result from the experiment is plotted in Fig. 7. Each set of the results is produced with three tries for each case. According to the analysis, when there are four objectives defined for the problem, the distance measurement in the RS (*Dist1*) and the worst-case is the lowest. However, the different is not so obvious compared to the result produced when the number of objectives defined equal to five. The distance measurement and the worst-case for five-objective problem is the highest. Meanwhile, in comparing the uniformity, when the number of objectives is three, the uniformity is the highest. Therefore, it can be concluded that the four-objective and three-objective problems give a more uniformed generating result and they produce a more acceptable solutions with lower distance in the solution set in the RS and with the better worst-case condition in comparison with the five-objective problem.

4 Conclusion

The state-of-the-art multiobjective metaheuristic is a practical solution for redistricting problem. This study has formulated the problem as a multiobjective one and has evaluated several important aspect of the design. This paper has presented the multiobjective metaheuristic design with its important components and functions. Several experiments also validate the method in the aspect of coverage of non-dominated solution, the tabu daemon with the ϕ-value and the preference of decision-maker with γ-parameter. When the -value considers the number of basic units, the result gives a better result of 3 % compared to the result without the consideration of the number of basic unit in the problem size. When the ϕ-value that considers the problem size and the number of desired output give

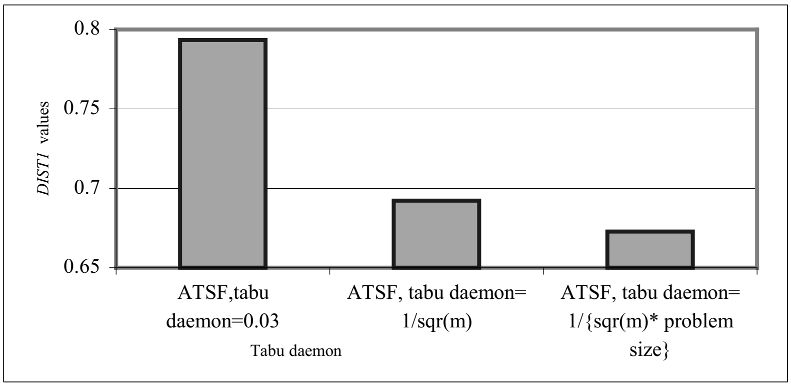


Fig. 5. *DIST1* for experiment with different ϕ-value

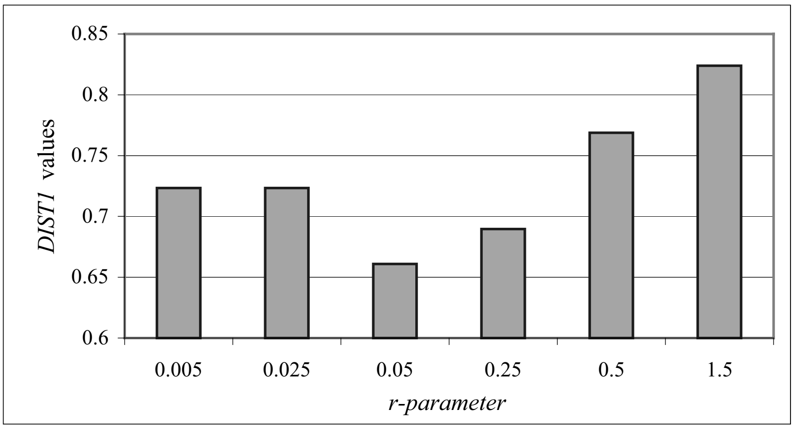
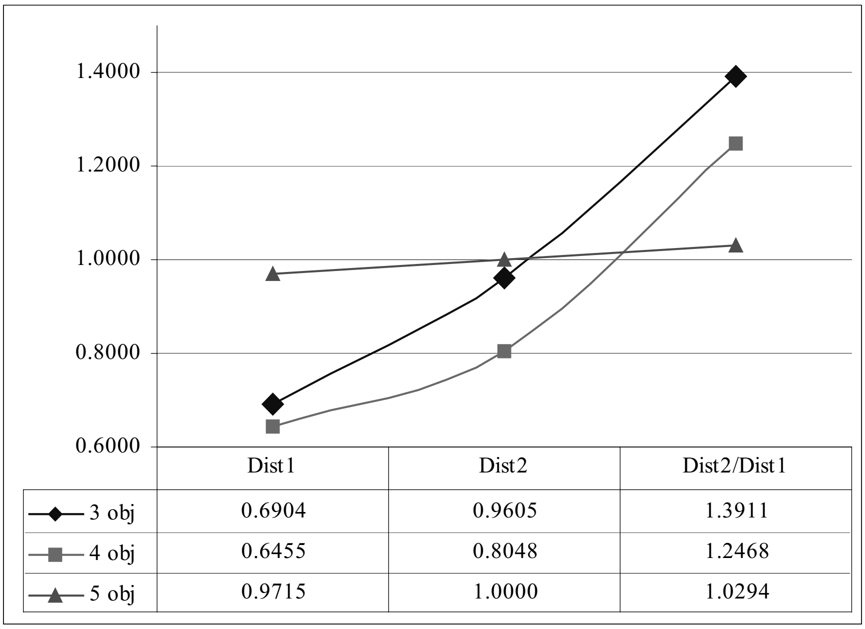


Fig. 6. *DIST1* for experiment with different γ-parameter



	Dist1	Dist2	Dist2/Dist1
◆ 3 obj	0.6904	0.9605	1.3911
■ 4 obj	0.6455	0.8048	1.2468
▲ 5 obj	0.9715	1.0000	1.0294

Fig. 7. Different Quality Measurement methods

the best result. On the other hand, the best value of the γ-parameter is equal to 0.05 because it produces the shortest distance in the measurement of the RS. Besides, the number of objectives defined is also evaluated for three, four and five objectives. When there are three objectives defined, the result performed is the less uniformed. Meanwhile, the four-objective and three-objective problems give a more uniformed result and they produce a more acceptable solution with lower distance in the solution set in the RS and with the better worst-case condition in comparison with the five-objective problem. In short, the proposed multiobjective metaheuristic is developed and evaluated in a systematic and scientific manner for redistricting. The design enables realistic problem definition with multiobjective definition and it produces a set of non-dominated solutions in a reference set.

5 References

[1]Agnew, J. (1994), "Geography and regional science program: geographic approaches to democratization," *National Science Foundation,* Washington.

[2]Alves, M.J. & Clímaco, M. (2000), "An interactive method for 0-1 multiobjective problems using simulated annealing and tabu search," *Journal of Heuristics* 6: 385–403

[3]Battiti, R. & Tecchiolli, G. (1994), "The reactive tabu search," *ORSA Journal on Computing* 6(2): 126-140

[4]Bong, C.W. and Wang, Y.C. (2004), "An intelligent GIS-based spatial zoning with multiobjective hybrid metaheuristic method," *Proceeding of The 17th International Conference On Industrial & Engineering Applications of Artificial Intelligence And Expert System (IEA/EIA 2004),* University of Otawa, Canada.

[5]Bozkaya, B., Erkut, E. and Laporte, G. (2003), "A Tabu Search Heuristic and Adaptive Memory Procedure for Political Districting," *European Journal of Operational Research 144*, pp.12-26.

[6]Czyzak, P and Jaszkiewicz, A. (1998), "Pareto simulated annealing–a metaheuristic technique for multiple objective combinatorial optimization," *Journal of Multi-Criteria Decision Analysis 7*, pp.34-47.

[7]Gandibleux, X. & Freville, A. (2000), "Tabu search based procedure for solving the 0-1 multiobjective knapsack problem: the two objectives case," *Journal of Heuristics* 6: 361-383

[8]Glover, F. and Laguna, M. (2002), "Tabu search," in Pardalos, P. M. and Resende, M. G. C. (eds.), *Handbook of Applied Optimization,* Oxford University Press, pp. 194-208

[9]Glover, F. (1998), "A template for scatter search and path relinking," in Hao, J.-K., Lutton, E., Ronald, E., Schoenauer M. and Snyers, D. (eds.), *Artificial Evolution, Lecture Notes in Computer Science*, 1363, pp.13-54.

[10]Hansen, M.P. (1997), "Tabu search for multiobjective optimization: MOTS," *Proceedings of MCDM'97*

[11]Jaszkiewicz, A. (2001), *Multiple Objective Metaheuristic Algorithms for Combinatorial Optimisation* (Draft), Habilitation thesis 360, Poznan University of Technology, Poznan.

Appendix: Algorithm for the proposed multiobjective metaheuristic

```
SEED ()
1.  Start
2.  Select a seed unit, x randomly
3.  Select adjacent basic units for x
4.  If number of districts is less than m
4.1     If total capacity size, P within interval [a, b]
4.1.1       Merge the basic units
4.2     Else
4.2.1       Reduce basic unit one by one until reach P
4.3     End if
4.4     Merge the left over basic unit with smallest district
5.  End if
6.  Check continuity
7.  Obtain seed solution, s
```

Fig. 8. Algorithm for seed solution initiator

```
GSC ()
1. Start
2. Gather district plan from SEED())
3. Group the basic units with their updated district number, where
       $x_{ij} \le x_{jj}, (i \in I, j \in J)$
4. Dissolve the basic unit layer to create a new district plan, $s_t$
5. If $s_t$ is not duplicate
5.1     Add $s_t$ to the generating solution
6. Else
6.1     Reject $s_t$
7. End if
8. End
```

Fig. 9. Algorithm for generated subset combination

```
NAT( )
1. Start
2. Get x with max($x_1$, $x_2$,... $x_n$)
3. For MOVE I
3.1     Select border unit of the x
3.2     Update attribute
4. End for
5. For MOVE II
5.1     Swap two border units, $x_m$ and $x_n$
5.2     Check tabu status
5.3     If not approve
5.3.1       Swap another two border units, $x_{m+1}$ and $x_{n+1}$
5.3.2       Store the border unit in tabu list
5.4     Else
5.4.1       Update attribute
5.5     End if
6.End for
7.End
```

Fig. 10. Algorithm for Neighbouring tabu move with tabu list and tabu daemon

```
Begin
SEED()
Set $RS=\varnothing$ and $\pi=1/n$ for all objectives j
Repeat for each boundary
   NAT()
   For each neighbour, x
      Get $\lambda$ from DM
      For each $x'$ where $f(x) \neq f(x')$
         Get $f(x')$ for all objectives
         If $x'$ where $f(x')$ is non-dominated by $f(x)$
            Measure the Manhattan distance norm
            Set distance proximity function
            For all objective j where $f_j(x) < f_j(x')$ Set
                  $\lambda_j = \lambda_j + \pi_j w$
         End if
      Loop
    Loop
    Normalise ($\lambda$)
    Calculate ATSF
    Select solution y with minimum ATSF and spatial continuity checking
    GSC()
    If y is non-dominated by any objective function in RS, then
       Update RS with y and Update $\pi_j$
    End if
  Loop
  Until for each boundary
End
```

Fig. 11. Procedure for hybrid metaheuristic when combined with MoSReF

Solving Facility Layout Problems with a Set of Geometric Hard-constraints using Tabu Search

Valdair Candido Martins[1] , Marco Antônio Barbosa Cândido[2],
and Leandro dos Santos Coelho[2]

Pontifícia Universidade Católica do Paraná
[1] *Programa de Pós-Graduação em Informática Aplicada*
[2] Grupo Produtrônica, Pós-Graduação em Engenharia de Produção e Sistemas
Rua Imaculada Conceição, 1155, CEP 80215-901, Curitiba, PR – Brazil
E-mail: {valdair; candido}@ppgia.pucpr.br; leandro.coelho@pucpr.br

Abstract. This paper approaches a computational solving of the industrial layout problem considering a set of hard-constraints not handled in previous works. The intention here has been to provide a new consistent benchmark problem that may help researchers to test their proposed algorithms. In order to handle the experiments, the computational optimization tool AVOLI (Visual Environment for Industrial *Layout* Optimization) is utilized. By using this computational tool, the provided hypothetic new facility layout problem mimicking a real one is solved in two steps: first a constructive-heuristic-based initial solution is generated and then Tabu Search (TS) heuristic is used to improve it.

Keywords: layout; optimization; hard-constraints; tabu search.

1 Introduction

As a direct consequence of the globalization a tough competition has been noted among the manufacture enterprises leading them to implement effective strategies targeting better quality and the increase of goods production. Surrounded by several activities is the planning of the facility layout under which the whole process of making goods takes place. The viable and efficient physical layout generation is a complex task (NP-Hard) [4]. It includes dealing with large amounts of data where human mistakes can result in a considerable increase of the production costs. Any organization, or any of its parts, can be viewed as a process. In very general terms, a process constitutes a transformation of inputs into outputs. On a high level of abstraction, a factory has three main areas of activities. It has, first of all, inputs, i. e., time, labor, money, and other resources. Inside the factory there are several transformation activities that take place, things that need to be done to produce the product or good. The process of transformation is carried out by a set of resources through a network of activities, and out of the factory comes the outputs. The whole purpose of working is to increase the quality and the quantity of the outputs by a very well management of the inputs and the internal services. We can assume the process of production of goods and services as being a number of operations performed by a group of suppliers, and a set of waiting times. Each supplier has it own process where the right accomplishment, under the time constraint, of the demanded tasks is the key activity. It is the interaction of the local services and supplier processes that determines facility efficiency. In this context, the development and application of simulation-based decision support tools for enhancing facility layout effectiveness is of paramount importance to optimize the arrangement of people and equipment on such environment. The physical layout design, in general, is difficult because there is no capable algorithm of finding an optimum solution for large scale problems in polynomial time. Real environments can have a large variety of facility areas that can be subject to a great number of constraints [15]. When design the background layout the facility layout designers must consider facilities planning such as the circulation area set-up, the moving "materials" system, the choice of where equipments and facilities will be sitting in a department or room, the location of input and output points, scheduling of services, inventory planning, ergonomic design, and so on. Before investing in a new facility or in redesign of an existing one, an organization should ensure that it makes the decision based on projected capacity requirements drawn from future market demand. Thus, process simulation is a valuable tool that could provide crucial information about these factors. As a core of many simulation tools, approaches such as Simulated Annealing – SA [10], Genetic Algorithms – GA [9],[8], and Tabu Search – TS [5],[6],[7] have been applied with promising results. Recent industrial applications using them are numerous and that subject has appeared in some articles of synthesis [12],[16]. Layouts can be built out of basic data, for instance, we have CORELAP [13], ALDEP [18], and PLANET [1], or improved from existing ones, as in the case of CRAFT [2], and COFAD [24].

This article is organized as following: In section 2 the scope of problem is discussed, in section 3 the adopted solution approach is explained, and in section 4 and 5, results and conclusion are presented.

2 Problem Scope

A solution is considered viable when it does not violate any of the following constraints:

- **Plant dimensions:** The space to accommodate all departments has a rectangular shape.
- **Area requirements:** Departments have unique area demands.
- **Shapes:** Departments are outlined as rectangular or square shape units.
- **Orientation:** Free (vertical or horizontal) or fixed (only vertical or only horizontal).

- **Aspect ratio:** The aspect ratio of the department *i* is defined as the quotient of the longer side divided by the shorter one. Departments with a variable aspect ratio allow flexibility on their rectangular shape.
- **Overlapping:** In single floor environments, departments cannot overlap.
- **Fixed and mobile:** A fixed department has its location known upfront. If a department is mobile, it can take any position into the external shape since it does not overlap any others.
- **Reduction ration:** A certain location, available to be assigned to a department, can't have size variation beyond imposed limits.
- **Not used areas (dead area):** (pillar, staircase, elevator, and so on). The existence of this areas can distort the rectangular shape of the useable area of department *i*. That possible loss will be diminished by the dead area ratio of the department *i*, defined as the as the quotient of the unused area within department *i* divided by the total area of department *i*.
- **Adjacency requirements:** In real life scenarios, departments must be located in the border of the plant, they can be adjacent to each other due to the interdependence among them, or they must not be adjacent. Departments are considered adjacent when their borders share at least two points.

The Table 1 shows the set of constraints handled by AVOLI [14] compared to those handled by Nugent *et al*. [17], Tam and Li [20], Tam [21], Suresh and Sahu [19], Tate and Smith [23], Tam and Chan [22], and Chiang [3].

Table 1. Set of constraints handled by AVOLI.

Constraints	Nugent	Tam	Tam	Suresh	Tate	Tam	Chiang	AVOLI
	1968	1991	1992	1995	1995	1998	2001	2002
External shape	x		x	x	x	x	x	x
Areas requirements		x	x	x	x	x	x	x
Shapes		x	x	x	x	x	x	x
Orientation (v/h)		x						x
Aspect ratio		x	x			x		x
Overlapping	x	x	x	x	x	x	x	x
Mobile departments	x	x	x	x	x	x	x	x
Fixed departments		x						x
Reduction ratio								x
Dead area			x					x
Adjacency requirements								x
Occupied areas			x					x
Problem size	30	30	30	30	30	20	40	55

2.1 Objective Function

In general, each department area, and the cost of interconnecting all pairs of departments is estimated from basic data collected by production engineers. This cost can be quantified in the weight function attributed to the material flow or it can be specified in qualitative terms through adjacency requirements. The aim is to minimize the cost of interconnecting fixed and mobile departments. In this case the objective function is given by the Equation (1).

$$\underset{s \in S}{Min} \quad F = \sum_{i}^{n} \sum_{j}^{n} C_{ij} F_{ij} D_{ij} \; M^{(k+l+m)} \tag{1}$$

i, j = 1,2, …, n, where:

- C_{ij}: flow cost between department i and department j by unit of distance.
- F_{ij}: flow between department i and j.
- D_{ij}: rectangular distance between centers of department i and j.
- K: number of departments not assigned to the layout).
- l: number of departments violating adjacency – close.
- m: number of departments violating adjacency – far.
- M: the positive value is large and defined as a function of the problem. If constraints k, l and (or) m, are violated, the evaluating objective function is highly penalized.
- n: size of the problem.

The aspect ration and dead area constraints are defined respectively by $a_{i(\min)} \leq a_i \leq a_{i(\max)}$ and $0 \leq \beta_i \leq \beta_{i(\max)}$, where:

- a_i : aspect ratio of the department i.
- $a_{i(\min)}$, $a_{i(\max)}$: minimum and maximum aspect ration of the department i.
- β_i , $\beta_{i(\max)}$: dead area ration and maximum dead area ration of the department i.

3 Solution Approach

The solution procedure starts with providing the specific input data for each problem. An allocation list that defines the department's allocation sequence in the bi-dimensional space is built taking into account the area and association among all departments. In the list, the largest area department is added first followed by one with which it is the most strictly associated. Subsequently, the next department to be inserted in the list is the one with the closest association with the last one inserted. If the last department has no association with another one, the next available department with the largest area is then inserted in. This procedure is repeated until all departments have been added to the list. When the list is built, the allocation process starts positioning the fixed departments in the layout followed by the mobile ones. The first department is allocated in the first available position into the external shape starting in the top-left corner of the layout. The second department is allocated in the first available position following a clockwise direction from the previously allocated department, and so forth. If the following element cannot be allocated in the vicinities of its predecessor respecting all constraints, then this department will be allocated in the next available location. This procedure is followed until all departments are allocated. In the case of a department is not allocated or if it violates some constraint, the procedure initiates again by rearranging the allocation list putting the second-largest department in the second position and shifting one position to the right the another elements. If in each attempt there is a violation, the list will be rearranged until all *n* possibilities have been tested.

3.1 Improvement on the Initial Solution

The TS approach is used as an improvement strategy for the initial solution. The basic parameters in TS are: 1) memory structure: it is used to store historic information on the evolution of the searching process, 2) aspiration criteria: it is used to balance a new configuration, based on the information stored in the memory structure, and 3) strategies of intensification and diversification: they are based on modifying choice rules to encourage move combinations and solution features historically found good. For a minimization problem, TS can be generalized in the following way:

1. Start from an initial solution *S*;
2. Parameters initialization: *TabuList* ← 0, *BestSolution* ← *S*;
3. WHILE a stop criterion is not reached DO: Evaluate the list of *PossibleMoves*. *PossibleMoves* = {*S'*∈ vicinity *V(S)*, *S'*∉ *TabuList* or *S'* pass over the aspiration criterion}: Select the best admissible solution, *S** ∈ *PossibleMoves*;
4. Update *TabuList* and *CurrentSolution* *S* ← *S**; IF *G(S)* < *G(Best)* Then *BestSolution* ← *S*;
5. Return *BestSolution S*.

General overview of the TS Utilized:

1. Problem representation: Allocation List (*n* departments).
2. Neighborhood *V(S)*: a) Simple swap, b) Pseudo simple swap, and c) Pseudo expanded swap.
3. Structure of control elements:
   - 3.1. Dynamic tabu list: a) Random tabu time, and b) variable tabu time.
   - 3.2. Aspiration criteria: a) A move leaves its tabu status if it generates a better solution than the best found one, b) If in the vicinity of the current solution all possible moves are tabu, the move that less worsen the objective function becomes free, or c) If doesn't occur any improvement on the objective function over *T* iteration, all tabu moves are set free.

   3.3. Stop criteria: Fixed number of iteration.

4. Solution decode: Departments allocation.

3.2 Tabu List Configuration

Tests were run using dynamic tabu list as: 1) random tabu time: in each interaction a random number is chosen within the specified timeframe defined by T_{min} and T_{max} and it is assigned to the element that has just became tabu; 2) variable tabu time: similar to random tabu time, but, if does not occur improvements on the objective function during *T* iterations, all elements in the tabu list are set free.

3.3 Neighborhood Structures

To illustrate how vicinity structures work, consider the list *L = (A, B, C, D, E, F)* of *n=6* departments to be allocated. Three neighborhood structures were used:

1. Simple swap *V(S)*. The possible switching moves are: *A* with *B*, *B* with *C*, *C* with *D*, *D* with *E*, *E* with *F*, and *F* with *A*. The best admissible move or a move that satisfy the aspiration criteria is chosen.
2. Pseudo simple swap: Similar to the simple swap, but, here the moves are not effectively carried out, they are only evaluated. All departments are considered to have the same shape and area, making it possible for the pseudo-switches. Since the flow relationship among departments doesn't change, what can really alter the results are distances and transportation cost among departments. This way, pseudo-distances can be used without the need to allocate departments to find the real distance among them. A subset $W \subset V(S)$ is tested, and then, finally the best admissible move is chosen to be done.
3. Pseudo expanded swap: *V(S)* is bigger. The number of neighbors is equal to $V*(V-1)/2$. The same logic of the pseudo simple swap is employed, with the only difference resting in the size of the vicinity being evaluated.

3.4 Strategy of Prohibition and of Liberation

The prohibition strategy manages what will be added to the tabu list and in what time it will occur by labeling as forbidden those moves recently done. After the occurrence of a move, the selected attributes that occur in solutions recently visited are tagged tabu-active, and solutions that contain tabu-active elements, or particular combinations of these attributes, become tabu. The liberation strategy manages what will be removed from the tabu list and in what time it will occur. Once one move becomes free its tabu restrictions are removed and that move is allowed to participate in future solutions. The aspiration criteria do not induce a particular move to be selected, but simply make it available, or alternately withdraw evaluation penalties attached to it.

4 Results

The experiments were handled by making use of the benchmark problems from Nugent *et al*. [17] and the AVOLI [14] software as a tool to validate the found results. The test list includes the problems with 5, 6, 7, 8, 12, 15, 20, and 30 facilities, with maximum of 500 iterations. The obtained results were compared to the results showed by Tate and Smith [23], Suresh and Sahu [19], Chiang [3], and to the best known solutions to Nugent *et al*. [17]. Table 2 shows our results for eight test problems along with the results reported by previous researchers. Although our best solutions did not improve upon the previously published best ones, with the TS approach, we reached very close results to those published solutions. The last column of Table 2 gives the distance in terms of percentages between the reached results and the best known benchmark solutions.

Table 2. Benchmark problems – Best solutions comparisons.

	P1	P2	P3	P4	P5	P6	P7	P8
Problem size	5	6	7	8	12	15	20	30
Max. num. of iteration	100	100	100	100	200	200	500	500
Total run time (s)	3	3	3	3	29	30	202	1159
Best iteration	8	1	23	8	39	120	300	171
Best time (s)	0	0	1	0	5	18	172	393
Nugent et al. (1968)	50	86	148	214	578	1150	2570	6124
Tate (1995)	50	86	148	214	578	1150	2598	6184
Suresh (1995)	50	86	148	214	578	1150	2570	6168
Chiang (2001)	50	86	144	212	578	1110	2564	6094
Avoli	50	86	148	214	578	1150	2570	6128
Distance (%)	0.00	0.00	2.70	0.90	0.00	3.47	0.23	0.55

A new benchmark problem enclosing 55 departments (50 moves and 5 fixed) under the set of realistic constraints showed in table 1 is utilized to test the AVOLI performance. This test environment encloses 5 dead areas, and the following restrictions: departments 8, 9, 19 and 29 must be in the vertical direction, departments 10 and 17 must be in the horizontal direction, department 9 must be close to 47, department 15 must be far from 19, department 23 must be far from the fixed department 3, department 14 must be far from the fixed department 4, and department 20 must be on the edge of the external shape.

The test was executed in a total of 500 iterations. The last row of Table 3 presents the percentage of improvement of the final solution in relation to the start one. The problem involves determining the location of machines, workstations, and other facilities to achieve the following objectives: 1) minimize the total costs, and 2) minimize the required area of the external shape. The best combination between neighborhoods and tabu time was detected when pseudo expanded swap *V(S)* and the structure of control elements, operated in the variable tabu time manner, was considered together. The right tuning of T_{min} and T_{max} allowed balancing the exploration of the solution space either by intensifying or diversifying when the attribute of a movement was strong restricted. The best gotten solution, when compared with the initial solution, reached 33.85% of improvement. It was observed that the most promising choice of the candidate list to participating in the exchanges serves to restrict the number of elements to be analyzed, therefore, a reduced neighborhood could be used without compromising the quality of the found solutions.

The input data for the considered problem are given in corresponding Tables 4 to 7. Figures 1 and 2 show the initial layout and the best found layout to the proposed problem, respectively. The departments are identified as following: the signal "-" and "+" proceeding the sequential numbers identifies fixed and move departments respectively. The occupied areas are identified sequentially. The tests were executed in a computer S3200NX Compaq with 256 MB of RAM.

Table 3. Experimental results for the proposed problem.

Problem size - External shape (m^2)	55 – 90X60
Variable tabu list - Expanded V(S)	T_{mim} (7) and T_{max} (14) – 20
Total of iterations - Total time (s)	500 – 12700s
Initial - Final solution cost	187770 – 124213
Best iteration - Best time (s)	216 – 5206s
Improvement (%)	33.85

5 Conclusion

In this paper, we proposed a new manufacturing layout problem of a hypothetic new facility that mimics realist ones. The suggested problem reveals the important role that process simulation can play in facility layout design since it encloses hard-constraints frequently found in real problems and not adequately handled in other models. The problem was solved in two steps: first a constructive-heuristic-based initial solution was generated and then tabu search with three neighborhood structures was applied in order to improve it.

Through the carried test we observed that the dimensions of the external shape has directly influence on the gotten results, therefore, its right adjustment can produce better solutions. The performance obtained by AVOLI in terms of efficiency is pretty good compared to other possible implementation solutions. From practical viewpoint, the complexity of the proposed problem was a very successful benchmark to AVOLI, once AVOLI could be validated through Nugent *et al*. [17] benchmark problems. Although the proposed problem was solved in acceptable computational time, it was not possible to make any comparison of results since similar problems had not been found in the literature. The tabu search adopted in this paper provides a way for quickly solving to optimality most of the layout problems analyzed. The strategy proposed in this work makes it straightforward to apply tabu search in AVOLI software to optimization of any layout model that can be specified in mathematical programming. In future works, other metaheuristics, such as improved ant systems and genetic algorithms will be included in AVOLI for performance and complexity computational comparative studies.

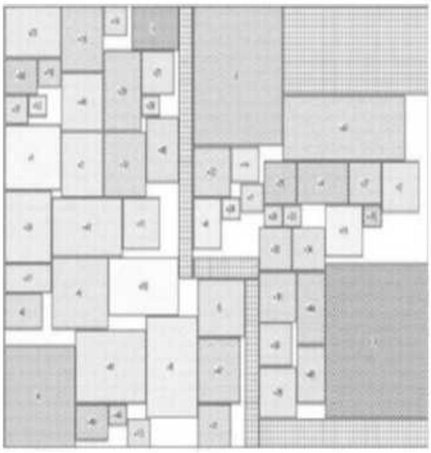


Fig. 1. Initial layout with 55 departments.

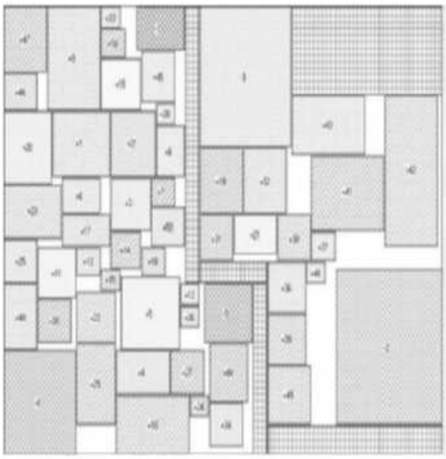


Fig. 2. Final layout with 55 departments.

Table 4. Geometry of the fixed departments.

	Area m^2	X initial	Y initial	Orientation	Dead area %	Min. Area Ratio	Max. Area Ratio
- 1	60	28	1	free	0	0.5	1
- 2	450	69	36	free	0	1	1
- 3	360	41	1	free	0	1	1
- 4	200	1	47	free	0	1	1
- 5	80	42	38	h	0	0.6	1.2

Table 5. Geometry of the occupied areas.

	X-initial	Y-initial	Width (m)	Length (m)
1	38	1	37	3
2	41	35	3	14
3	60	1	12	31
4	52	38	23	3
5	55	57	4	36

Table 6. Flow relationship among departments.

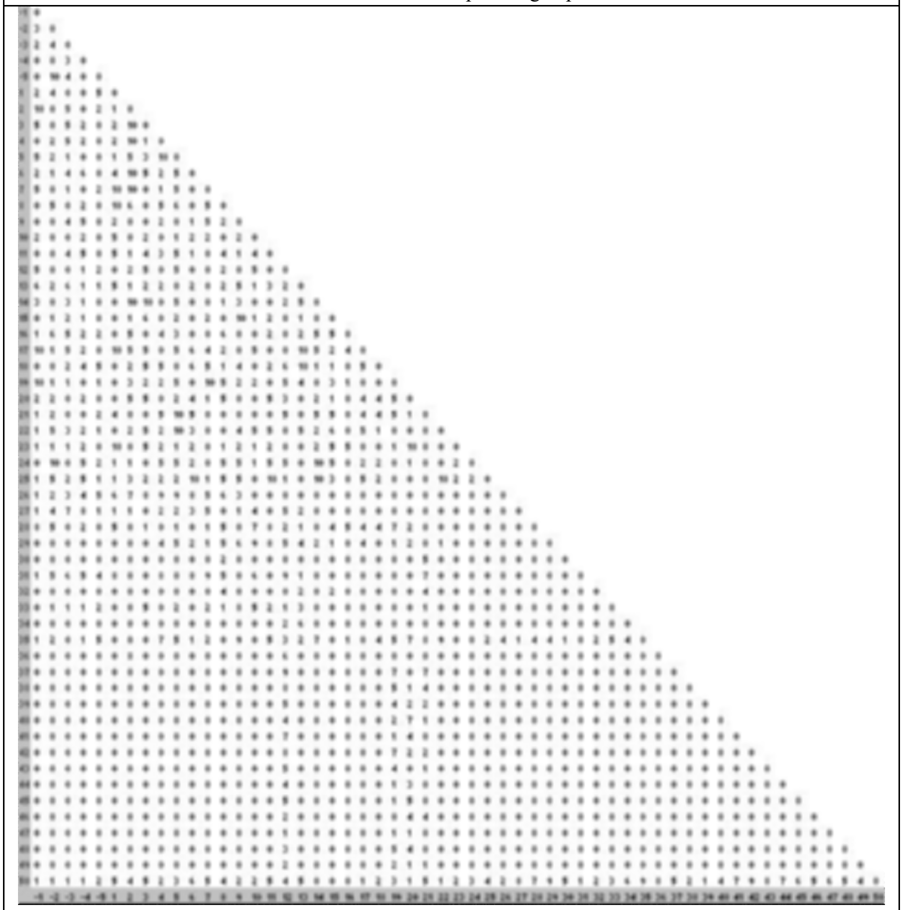
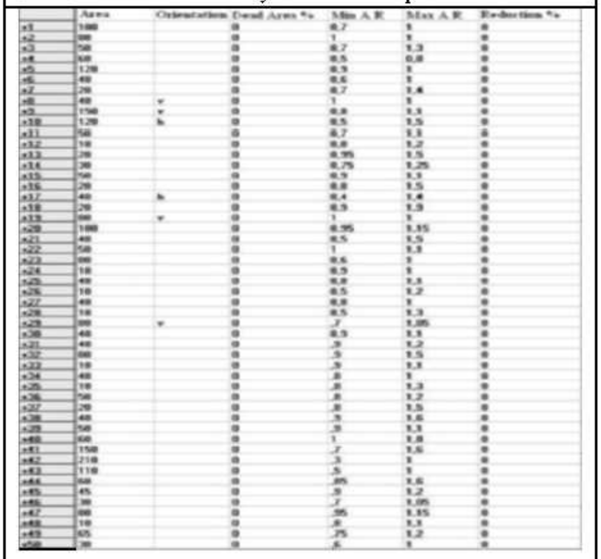


Table 7. Geometry of the move departments.



	Area	Orientation	Dead Area %	Min A.R	Max A.R	Reduction %
x1	100		0	0,7	1	0
x2	80		0	1	1	0
x3	50		0	0,7	1,3	0
x4	60		0	0,5	0,8	0
x5	120		0	0,5	1	0
x6	40		0	0,6	1	0
x7	20		0	0,7	1,4	0
x8	40	v	0	1	1	0
x9	150	v	0	0,8	1,1	0
x10	120	h	0	0,5	1,5	0
x11	50		0	0,7	1,1	0
x12	10		0	0,8	1,2	0
x13	20		0	0,95	1,5	0
x14	30		0	0,75	1,25	0
x15	50		0	0,9	1,1	0
x16	20		0	0,8	1,5	0
x17	40	h	0	0,4	1,4	0
x18	20		0	0,5	1,9	0
x19	80	v	0	1	1	0
x20	100		0	0,95	1,15	0
x21	40		0	0,5	1,5	0
x22	50		0	1	1,1	0
x23	80		0	0,6	1	0
x24	10		0	0,5	1	0
x25	40		0	0,8	1,1	0
x26	10		0	0,5	1,2	0
x27	40		0	0,8	1	0
x28	10		0	0,5	1,3	0
x29	80	v	0	,7	1,05	0
x30	40		0	0,5	1,1	0
x31	40		0	,9	1,2	0
x32	80		0	,5	1,5	0
x33	10		0	,9	1,1	0
x34	40		0	,8	1	0
x35	10		0	,8	1,3	0
x36	50		0	,8	1,2	0
x37	20		0	,8	1,5	0
x38	40		0	,5	1,6	0
x39	50		0	,9	1,1	0
x40	60		0	1	1,8	0
x41	150		0	,7	1,6	0
x42	210		0	,3	1	0
x43	110		0	,5	1	0
x44	60		0	,85	1,6	0
x45	45		0	,9	1,2	0
x46	30		0	,7	1,05	0
x47	80		0	,95	1,15	0
x48	10		0	,8	1,1	0
x49	65		0	,75	1,2	0
x50	30		0	,6	1	0

Acknowledgments

This paper is supported in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – CAPES under grand BEX1152025.

References

1. Apple, J.M. and Deisenroth, M.P. (1972) A computerized plant layout analysis and evaluation technique PLANET, *Proceedings of the 23rd Annual Conference and Convention, AIIE, CA*, pp. 121-127
2. Armour, G.C. and Buffa, E.S. (1963) A heuristic algorithm and simulation approach to relative allocation of facilities, *Management Science* 9, pp. 294-309
3. Chiang, W.C. (2001) Visual facility layout design system, *International Journal Production Research* 39(9), pp. 1811-1836
4. Garey, M.R. and Johnson, D.S. (1979) Computers and intractability: a guide to the theory of np-completeness, *W. H. Freeman and Company*, New York
5. Glover, F. (1989A) Tabu search - part I, *ORSA Journal on Computing* 1(3), pp. 190-206
6. Glover, F. (1989B) Tabu search - part II, *ORSA Journal on Computing* 2(1), pp. 4-32, 1989b
7. Glover, F. and Laguna M. (1997) Tabu search, *Kluwer Academic Publishers*. Massachusetts

8. Goldberg, D.E. (1989) Genetic algorithms in search, optimization and machine learning, Addison Wesley, New York, NY, USA
9. Holland, J.H. (1975) Adaptation in natural and artificial systems, Univ. Michigan, Ann Arbor, MI
10. Kirkpatrick, S.; Gellat, C. D. JR, and Vecchi, M. P. (1983) Optimization by simulated annealing, *Science* 220, pp. 671-680
11. Koopmans, T.C. and Beckman, M. (1957) Assignment problems and the location of economic activities, *Econometrica* 25(53), pp. 53-76
12. Kusiak, A. and Heragu, S.S. (1987) The facility layout problem, *European Journal of Operational Research* 29, pp. 229-251
13. Lee, R.C. and Moore, J.M. (1967) CORELAP - Computerized relationship layout planning, *Industrial Engineering* 18, pp. 195-200
14. Martins, V.C. (2002) Visual Environment for Industrial *Layout* Optimization - AVOLI industrial model, *Patent at INPI*, Register Number.00048682 - Brazil
15. Martins, V.C., Coelho, L.S., et al. (2003) Industrial layout optimization based on tabu search, *Gestão & Produção* 10(1), pp. 69-88. ISSN 0104-530X (in Portuguese)
16. Meller, R.D. and Gau, K.-Y. (1996) The facility layout recent and emerging trends and perspectives, *Journal of Manufacturing Systems* 15(5), pp. 351-366
17. Nugent, C.E., Vollmann, T.E. and Ruml, J. (1968) An experimental comparison of techniques for the assignment of facilities to locations, *Operations Research* 16, pp. 150-173
18. Seehof, J.M. and Evans, W.O. (1967) Automated layout design program, *Industrial Engineering* 18, pp. 690-695
19. Suresh, G. and Sahu, S. (1995) A genetic algorithm for facility layout, *International Journal of Production Research* 33(12), pp. 3411-3423
20. Tam, K.Y. and Li, S.G. (1991) A hierarchical approach to the facility layout problem, *International Journal of Production Research* 29, pp. 165-184
21. Tam, K.Y. (1992) Genetic Algorithms, function optimization, and facility layout design," *European Journal of Operational Research* 63, pp. 322-346
22. Tam, K.Y. and Chan, S.K. (1998) Solving facility layout problems with geometric constraints using parallel genetic algorithms: experimentation and findings, *International Journal of Production Research* 36(12), pp. 3253-3272
23. Tate, D.M. and Smith, A.E. (1995) A genetic approach to the quadratic assignment problem, *Computers and Operations Research* 32, pp. 73-83
24. Tompkins, J.A. and Reed J.R. (1976) An applied model for the facilities design problem, *International Journal of Production Research* 14(5), pp. 583-595
25. Vollman, T.E. and Buffa, E.S. (1966) Facilities layout problem in perspective, *Management Science* 12, pp. 450-468

Part VII

Complexity Management

Empathy: A Computational Framework for Emotion Generation

Graziano Catucci, Fabio Abbattista, Rita Chiarella Gadaleta,
Domenico Guaccero, and Giovanni Semeraro

Università degli Studi di Bari
Via Orabona 4, 70125 Bari, Italy
catucci@di.uniba.it, fabio@di.uniba.it, chiarella.rita@webmail.it,
guaccero@di.uniba.it, and semeraro@di.uniba.it

Summary. Empathy is a distributed environment for the generation of emotions and other related affective phenomena like moods and temperaments. Empathy has been conceived as an object-oriented reusable framework entirely written in Java and realized for the purpose of studying the direct influences of emotions on behaviors and on decision-making processes of autonomous agents, interacting in complex or real environments. It allows for the realization of custom emotional agents, usable in several different domains, from the educational applications (e.g. entertainment, video games, intelligent tutoring systems ...) to control systems in autonomous robots.

Key words: autonomous agents, emotional agents, synthetic characters

1 Introduction

Emotions have been often misconceived by the "official" occidental culture that has confined them to the pure subsidiary role of sustaining reason and elevated cognitive processes. Consequently, for a very long time, emotions have been seen as a disturbing element of abstract reasoning and as a not desirable property for an intelligent artificial system [9]. In this manner can be briefly summarized the vision of classic Artificial Intelligence, whose spirit is exemplary personified by J. McCarthy in his amusing short story "The Robot And The Baby" [10].

A complete revalutation of emotion and correlated complex affective phenomena is due to the most recent research in neurobiological and psychological fields, thanks mainly to the work of researchers like Damasio, LeDoux [3, 7] and Izard [6].

Following the approach delineated by their perspective, Empathy implements an Emotion Generation System that takes in full account the most recent discoveries in emotion field. In particular, it addresses topics like the influencing of emotions on behavior and its relation with decision-making affected activities. Differently from the major part of emotion applications developed in the last twenty years and dealing with the improvement of human-like interfaces [8, 13, 1], Empathy considers not only the cognitive aspect of emotion activation, but also a set of different factors certainly involved in emotion processes like drives and neurotransmitters.

Empathy presents a new model for emotions representation and generation, adopting the solutions offered by the Cathexis model [15], realized by J. Velásquez at the Massachusettes Institute of Technology (MIT), and enriches it by taking into consideration several new factors, among which the most important are: The somatic markers, as hypothesized by Damasio in [3]; an intrinsic unconscious factor proper of emotional phenomena; finally, a motor system represented by a 3D facial animation engine [16].

The particular way in which Empathy has been conceived, a reusable object oriented framework implemented in the Java programming language, makes possible to adapts it to a wide variety of contexts and software architectures. Autonomous agents and control systems embedded in autonomous robots are two of possible domains expressly addressed by this framework.

2 The Empathy Model

An accurate description of the Empathy model requires a close look at the main subject of this work: Emotions and affective phenomena. This led us to the difficult task of giving them an appropriate definition that would be able to reflect the main properties that have been ascribed them in the related literature. There is some agreement among the researchers on the basic components of emotion. Elicitors or activators are a fundamental one and they can be classified in several types, among which *cognitive elicitors*, *neural elicitors*, *motivational elicitors* and *sensorimotor elicitors*. Expressive and motor components constitute an other point of convergence and some of the aspects involved with them include central nervous system efferent activity, prototypical facial expressions, body posture, head and eye movements, vocal expression, and muscle action potentials. Naturally, there is no objection against the consolidated fact that emotion, once generated, is registered in the consciousness. This constitutes what is commonly called *emotion experience* or *feeling* and it represents often an obstacle to the study of emotion. Now, we can proceed to present how the approach considered in this work models the various aspects of an emotion and which of the various characteristics considered above are really represented in the model.

Empathy provides explicit representations only for the six basic emotions such as they have been individuated by Paul Ekman in his work on facial expressions [5]: *Anger*, *Fear*, *Sadness*, *Disgust*, *Joy* and *Surprise*. Each of them is designated by a protospecialist of mind, according to the "Society of Mind" vision of Marvin Minsky [11]. A prototospecialist is an elementary entity, a sort of a building-block of mind cognitive processes that Minsky utilizes for the description of the various mental activities. All other affective phenomena, like blend emotions, mixed emotions, moods and temperaments originate as emergent phenomena from the interaction of these basic emotions. Empathy models all affective processes through a distributed environment of six interacting protoagents, each representing a primary emotion, a term used by Damasio to indicate basic emotions and adopted in this work to distinguish them from the more complex secondary emotions that involve emotional memories [3] [4].

Figure 1 (adapted from [15]) reports the general architecture of the Empathy model in which it is possible to detect the relationships between the "Emotion Generation System", the "Emotional Behavior System" and the "Motor System". The first one is responsible for the generation of the emotional state of the system relying on the internal motivations and external environment events. The second one uses this emotional state to select the appropriate behavior consistent with the actual state of the agent and with the previous past experience. Finally, the "Motor System" reports the current Emotional Expression representing it as a 3D face animation constructed by the FACE (Facial Animation Compact Engine) system [16].

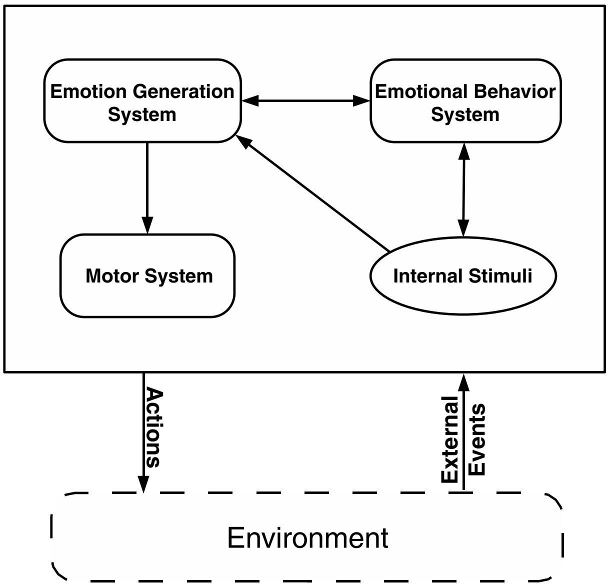


Fig. 1. Empathy Architecture

3 Emotions and Affective Phenomena

The "Emotion Generation System" is composed of six emotional protoagents, one for each fundamental emotion family: namely fear, anger, disgust, sadness, surprise and joy. They are fully interconnected with inhibitory and excitatory gains that let them to affect other emotion protospecialists and to prevent what Minsky called the "Avalanche Effect", a situation in which all protospecialists influence each other leading to an explosion of several emotions active at the same time. Each protospecialist is equipped with four different sensors direct matching to the four distinct emotion activation systems reported by Izard in [6] and visible in Figure 2 (adapted from [6]). The *"Neural System"*, independently from the other protospecialists, is able to activate the various emotional processes and is always running. The *"Sensorimotor System"* provides facial expression feedback based on the current emotional expression represented by the system. The *"Motivational System"* comprehends both drives (*Fatigue*, *Interest*, *Hunger*, ...) and emotions as causal factors of emotion and affective episodes. The *"Cognitive System"* takes into consideration the evaluation of external events and their cognitive valence for the agent experiencing a particular combination of events. This subsystem has been implemented in compliance with the Roseman's theory [14] which provides a direct causal relationship between specific appraisal configurations and the corresponding emotional state. Roseman takes into consideration a greater number of discrete emotions than that reported in Empathy, so a simplification of the model has been made to be adapted to our model. An other important question that has led us to this particular choice is that also Izard states that Roseman's theory fits well in his emotional multilevel activation system, because it contemplates the weight of motivational factors into emotion activation. All these subsystems are responsible for the activation of emotional processes, but their involvement varies according to different weights and priorities, and specifically the exact order is *neural*, *motivational*, *sensorimotor*, and *cognitive.* It is extremely noticeable the fact that the cognitive aspect is the last in order, contrary to the most part of existing emotional systems.

Let see in more details the characteristics of the "Protospecialist of Emotion". There are five main elements to consider in the characterization of a single emotion protoagent, the α and ω thresholds, the intensity decay function ϕ, the emotional intensity I, and the uncertainty U. The intensity I measures the emotional excitement and it is represented as a float positive value. The α threshold sets an inferior limit for the particular emotional intensity, above which the correspondent protoagent activates itself. The term *"activation"* is used here whith a precise, particular meaning: A protospecialist is active when it releases its emotional intensity to the "Emotional Behavior System" and to the others emotion protospecialists. So, this means that a protoagent has always a low level of arousal even if it isn't active, a state that leads to a natural way of modeling moods. The ω threshold determines the superior limit for the emotional intensity. It is the maximum arousal allowed for it, a boundary above which the emotional experience has no relevance for the peculiar emotional agent modeled by the user of the Empathy framework. The specific value of the two thresholds lets the agent designer to implement certain characteristics of the emotional agent, like its temperament. The decay function ψ treats an argument often not considered by several emotional theories and software systems considered in the past. It can be implemented, and it is the case of Empathy, as a simple linear function or as a more complex function of the time. It affects the emotional intensity by decreasing its value if no external or internal events happen. This behavior is in perfect accordance with real life observations around emotional episodes whose effects vanish if the generating causes go less. The level of uncertainty U has been introduced to take in account the unconscious factors always involved in every emotional phenomena and the capacity of anticipating future situations.

Empathy gives extreme flexibility to manipulate all these emotional attributes and uses the classical object-oriented mechanisms of inheritance and polymorphism to make possible the creation of custom emotional agents that present specific values for the two threshold, α and ω, for the level of uncertainty U, and a decay function ψ that differs for each emotion protospecialist. This specific approach allows us to model a potential infinite set of emotional agents with different temperaments, even emotional disorders.

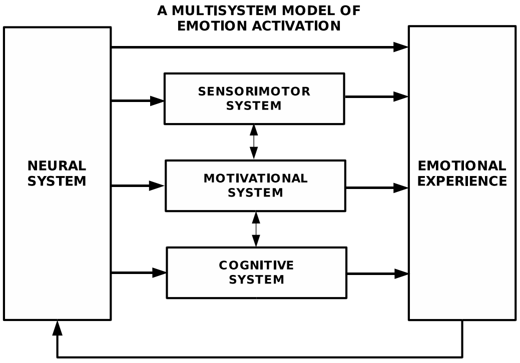


Fig. 2. Four Emotion Activation Systems

3.1 Emotion Blends and Mixed Emotions

Following Ekman, as described above, we have given explicit representation for a discrete number of six basic emotions. The particular choice is dictated not only by the according between facial expressions and emotional state, but also by the particular interpretation given by Ekman to them. These basic or primary emotions have been selected by evolution to confer to natural beings the capacity to deal with fundamental life tasks, such as avoiding danger, fighting, ...

More complex affective states like *emotion blends* and *mixed emotions* are obtained in Empathy through the activation and the interaction of two or more protoagents. Emotion blends are given by the co-occurrence of protoagents belonging to the same emotional family, while mixed emotions regard contrasting families. It is possible to be happy and sad at the same time, e.g. if the favorite team looses, but we win a bet on this game result.

3.2 Emotion Intensity

In this Section we give a formal presentation of the emotional intensity discussed above and we illustrate how the described several factors affect its value.

The emotional intensity, in Empathy, is affected by its previous level of arousal (which includes the mood), by the intensity of the other protospecialists (with inhibitory and excitatory gains) and by the different elicitors influencing that particular emotion protospecialist. This description is formally summarized by equations 1 and 2,

$$EI_{it} = \chi\left(\psi\left(EI_{i(t-1)}\right) + \sum_k EL_{ki} + \sum_l \left(EX_{li} \cdot EI_{lt}\right) - \sum_m \left(IN_{mi} \cdot EI_{mt}\right), 0, \omega_i\right) \tag{1}$$

$$\chi(EI) = min(\omega, max(0, EI)) \tag{2}$$

where EI_{it} is the value of the intensity of emotion i at time t; $EI_{i(t-1)}$ is its value at the preceding instant time $t-1$; $\psi()$ is the decay function for the emotion i; EL_{ki} is the emotion elicitor k for k that belongs to the set of emotion elicitors for emotion i; EX_{li} is the excitatory gain that emotion l applies to emotion i; EI_{lt} is the intensity of emotion l at time t; IN_{mi} is the inhibitory gain that emotion m applies to emotion i; EI_{mt} is the intensity of emotion m at time t; χ is the constrain function (equation 2) used to limit the emotion intensity i at time t, EI_{it}, in the range between 0 and ω, because, as stated early, there is no need to let the protoagent to go above a certain level of saturation.

4 Emotional Behaviors

In the last Section the “Emotion Generation System” has been described, in this Section we analyze how the “Emotional Behavior System” selects the appropriate behavior in accordance with the actual emotional state of the system. This issue regards how to decide and model the influences of emotions on the system once they have been activated. Experience and most theoretical studies state that the emotion experience has powerful influences in memory processes, goal generation, action selection, further generation of emotions, reasoning style, and learning. This is really a very difficult task to achieve, especially if we think to the potentially infinite set of combinations of all these factors. However a good model of emotional phenomena must comprehend some of them and must be open enough to allow the future inclusion of other ones inside it. This is the particular approach chosen in the design of the Empathy architecture.

The "Emotional Behavior System" is a distributed system of possible behaviors that compete for the control of the system. The choice of a particular behavior is based on the value of each possible behavior, which is determined at every update cycle of the system. Two are the main components of the system: The "Expressive Component" and the "Experiential Component" (see Figure 3, adapted from [15]).

Several researchers, among which Izard and Plutchik [6, 12], agree that emotions and moods influence directly facial expression and body posture. So the Experiential Component takes into consideration the following aspects:

- *Prototypical facial expression*, provides motor commands to alter the facial expression;
- *Body posture*, a set of motor commands to change the current body posture conforming to the actual emotional state;
- *Vocal expression*, modifies the corresponding expression varying some parameters like loudness, pitch, and temporal variability.

Actually, Empathy implements only the first one of these aspects, but it is generally enough to contemplate the future inclusion of the other two aspects.

The experience of emotions and moods affect also, as just observed, cognition and action and it is possible to distinguish two main characteristics:

- *Motivation*, a certain behavior, when active, may influence a particular motivation. For example, *eating* and *drinking* decrease respectively the level of hunger and thirst.
- *Action tendency*, this aspect is the most evident in emotional responses and it is realized in Empathy as a direct response to the triggering emotional event.

4.1 Behavior Selection

The selection of a particular behavior has been inspired by the work of ethology and in particular by Blumberg one which considers also the field of computer animation [2]. Particular conditions, events and objects relevant for a certain behavior are all represented by *releasing mechanisms* (releasers), a useful abstraction that permits to monitor any change happening in internal and external environment. The releasers are implemented in Empathy as real

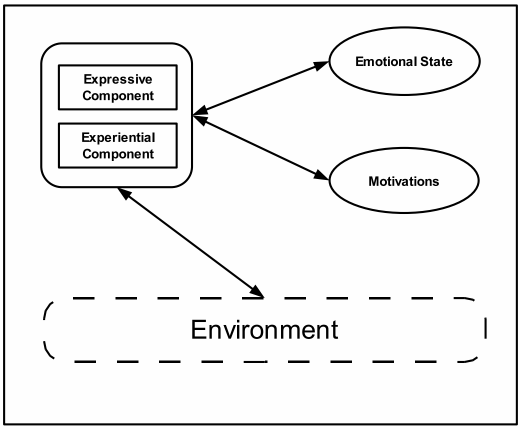


Fig. 3. Empathy, the Emotional Behavior System

values, whose meaning is dictated by a specific behavior to which they are associated.

The selection of a particular behavior represents a fundamental distinction point between Empathy and Cathexis, because in our model a new factor, the *somatic marker*, is considered. In certain emotional situations, characterized by high levels of intensity of a specific set of emotions all strongly correlated (like anger and fear, or joy and surprise), the selected behavior is opportunally *marked* by a positive or a negative value and it is put in relationship with the current emotional state of the system. This link between the behavior and the state of the system is implemented in Empathy as a couple of values *(Emotional State, Somatic Marker)* attached to the various behaviors supplied by the emotional agent. This information will be take into consideration by the "Emotional Behavior System" only when the system will set itself in the same emotional state that originated the somatic marker. This interpretation of the somatic marker is fully compliant with that given by Damasio [3] because it is put in relationship with both behavior and uncertainty factor (see equations 3 and 4), which, as early stated, symbolize the capacity of the agent to anticipate future behaviors on the base of past experience. Formally, the overall process can be summarized as follows:

$$EB_{it} = \begin{cases} \sum_k BR_{ki}, & \text{if} \quad ES_t \neq ES_{t_i} \\ \sum_k BR_{ki} + \sum_l SM^+_{li} - \sum_m SM^-_{mi}, & otherwise \end{cases} \tag{3}$$

and

$$ES_t = \sum_i U_i \cdot EI_i \tag{4}$$

where EB_{it} is the value of the emotional behavior i at time t; BR_{ki} is the value of the releaser k relevant to the behavior i at time t with k varying on the set of releasers meaningful to the behavior i; ES_t is the value of the emotional state at instant time t; U_i is the uncertainty factor associated to the emotional protoagent i; ES_{t_i} is the emotional state of the system at time t_i denoting the past instant time in which the somatic marker has signed the corresponding behavior; SM^+_{li} and SM^-_{mi} are respectively the values of the positive and negative somatic markers with l and m that vary on the set of markers relevant to the behavior i.

5 Cindy

The Empathy framework has been used to implement Cindy [16], an emotional 3D agent representing a young virtual girl immersed in a hypothetical environment properly selected to test Empathy. It provides several physical objects and gives the possibility to the user to manipulate them through a set of predefined actions. The user can interact with Cindy by modifying certain parameters of the internal state of the system, such as the internal motivations (interest, fatigue, thirst, hunger, . . .) and the properties of each protospecialist (activation threshold, saturation threshold, level of uncertainty, . . .).

This choice has been dictated by the will of realizing an ideal environment to experiment the possibilities offered by the Empathy framework and to properly set the characteristics of the emotional agent that the developer would modeling.

Cindy has been provided with a basic set of behaviors, some very general like *eating*, *drinking*, *reading*, . . . and other more specific behaviors like *watching a movie*, *reading a book*, *playing music*, *listen to music*.

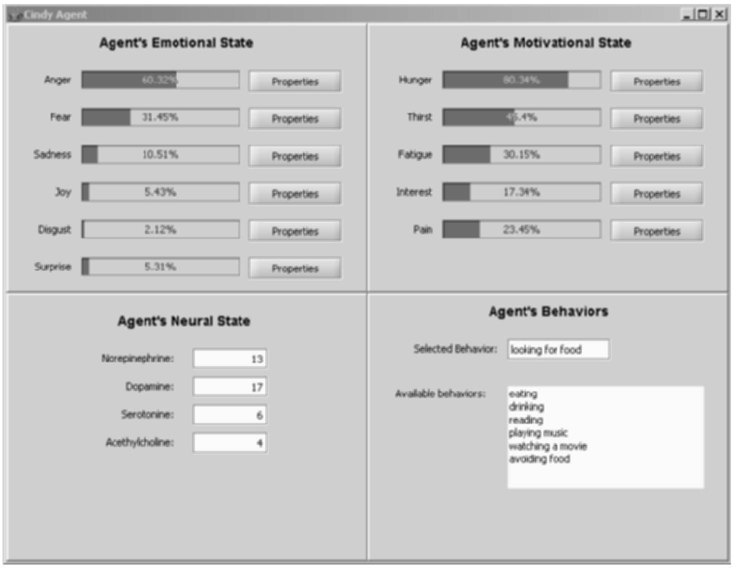
The categorization of the various behaviors in two distinct classes, each with own priority, even if not explicitly represented, lets to analyze the behavior of Cindy in particular situations. For example, if the intensity levels of *fatigue* and *interest* are high, then it is possible to observe what will be the next behavior selected: *sleeping* which is the more specific and appropriate or the more general *reading a book*.

Each *Emotional Behavior* has its own facial expression and experiential component, exactly as described in Section 4. The first one is directly related to the intensity of the various protospecialists. The second one includes all the influences of the behavior actually selected on the agent's motivations. For example, *"Eating"* and *"Drinking"* have experiential components that decrease the intensity values of hunger and thirst internal motivations.

In Figure 4 is reported a typical experimental scenario in which the user has inserted the values of the various neurotransmitters that influence the six basic emotions and the corresponding emotional and motivational intensities. The relative facial expression of Cindy is shown in Figure 5. In this case the basic emotions *Anger* and *Fear* present values superior to that of all other basic emotions and the same happens for two fundamental motivations, *Hunger* and *Thirst* that are very citical for the survival of the agent in its environment.

The selected behavior *"looking for food"* is not able, for the absence of food and water and in the environment, to satisfy the drives mentioned above and so the emotional state induced is a blend of fear and angry.

The mechanism of somatic marker implemented in Empathy cannot be, at this early stage of development of the framework, fully tested because Empathy lacks some fundamental aspects like emotional memories and secondary emotions which are necessary to store and retrieve past emotional experiences. This results in a limited study of the influences of emotions on behaviors and on decision-making processes.



Agent's Emotional State

Anger 60.32%
Fear 31.45%
Sadness 10.51%
Joy 5.43%
Disgust 2.12%
Surprise 5.31%

Agent's Motivational State

Hunger 80.34%
Thirst 45.4%
Fatigue 30.15%
Interest 17.34%
Pain 23.45%

Agent's Neural State

Norepinephrine: 13
Dopamine: 17
Serotonine: 6
Acethylcholine: 4

Agent's Behaviors

Selected Behavior: looking for food

Available behaviors:
eating
drinking
reading
playing music
watching a movie
avoiding food

Fig. 4. The client application which lets user to modify Cindy's internal state

5.1 Implementation details

Implementing Cindy or any other emotional agent is a task consiting of a well defined set of steps. The most fundamental one is the *"designing phase"* that is relatively to the choice of the particular agent behaviors, motivations and other details like that concerning activation and saturation thresholds, decay function, and so on . . . Successively, the agent implementation is realized as a class which encloses all the properties and the characteristics previously selected at design time. In particular, this is achieved by supplying it with a set of attributes like *Behaviors* and *Motivations* that are implemented with the classical *list* and *dictionary* data structures. An other important attribute is represented by the *Blackboard*, a global shared memory which lets the communication between the various subsystems. Strictly associated with these

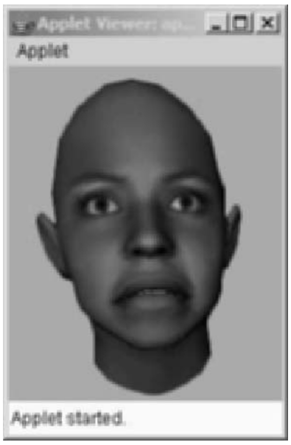


Fig. 5. Cindy's facial expression

attributes are methods to dynamically add, remove and/or update the corresponding *Drives*, *Emotions*, and *Behaviors*.

Finally, it is possible to represent the core part of the Cindy agent as an algorithm for emotion generation and action-selection. An overall description is given by the following (adapted from [15]):
On each update cycle:

1. Any external event is revealed and posted on the blackboard as an *environment event*.
2. Motivations values are updated. This is achieved in two different steps:
   a) Each *drive protospecialist* updates itself as follows:
      - Growth and damp rates are applied to the intensity value of the current drive.
      - Behavioral influences, if present, are applied to the current drive.
      - Intensity variation is a specific event that is posted on the blackboard as a *motivational event*.

   b) Each *emotion protospecialist* update itself as follows:
      - If the current emotion protospecialist is active, its *decay function* is evaluated.
      - All *emotion elicitors* of the current protospecialist are avaluated on the ground of the different events present in the blackboard.
      - Inhibitory and excitatory gains are added to the intensity of the actual emotion.
      - The intensity of the emotion protospecialist is constrained in the interval between 0 and the saturation threshold ω.
      - Expressive component is updated according to the values of the six basic emotions.

- If the intensity is greater than the activation threshold α, then its value is released to the appropriate emotion protospecialists, inhibiting or exciting them.
- The intensity variation is posted as an *emotional event* on the blackboard.

c) All emotional behaviors are updated relying on sensor input, both external or internal, and on the value of the *somatic markers* if the same behavior also happened in the past.

d) The emotional behavior with the highest value is selected and a new behavioral event is generated.

e) The *expressive component* of the selected behavior is generated, if this last possesses one.

f) The *experiential component* of the selected behavior is generated, if this last possesses one.

6 Conclusions and Future Work

Empathy lets to address the most part of affective phenomena and takes into account both cognitive and non-cognitive elicitors of emotions. Nevertheless, certain more complex aspects as secondary emotions and emotional memories have not been implemented.

It is a deliberate choice that has led to the design of Empathy as an open software architecture that lefts opened several implementation opportunities to expand its core. One of the most important is the enhancement of the *Emotional Behavior System* which would be realized as a distributed network of behaviors on different layers and each with its own priority. The first further improvement should be the inclusion of emotional memories and secondary emotions, absolutely necessary in order to add to Empathy more complex cognitive faculty like learning.

References

1. J. Bates, B. Loyall, and S. Reilly. An Architecture for Action, Emotion, and Social Behavior. Technical Report CMU-CS-92-144, School of Computer Science, Carnegie Mellon University, 1992.
2. Bruce Blumberg. Action Selection in Hamsterdam: Lessons from Ethology. In *Third International Conference on the Simulation of Adaptive Behavior*, pages 108–117, Brighton England, 1994. MIT Press.
3. Antonio R Damasio. *Descartes' Error. Emotion, Reason and the Human Brain.* New York: Gosset/Putnam, 1994.
4. Antonio R. Damasio. *The Feeling of What Happens. Body and Emotion in the Making of Consciousness.* Harcourt, 1999.
5. Paul Ekman. Basic Emotions. In T. Dalgleish and M. Power, editors, *Handbook of Cognition and Emotion.* John Wiley & Sons, 1999.

6. Carroll E. Izard. Four Systems for Emotion Activation: Cognitive and Noncognitive Processes. *Psychological Review*, 100(1):68–90, 1993.
7. Joseph LeDoux. *The Emotional Brain*. New York: Simon and Schuster, 1996.
8. A. B. Loyall and J. Bates. Real-time Control of Animated Broad Agents. In *Proceedings of the Fifteen Annual Conference of the Cognitive Sciences Society*. Boulder, 1993.
9. John McCarthy. Making Robots Conscious of their Mental States. *http://www-formal.stanford.edu/jmc*, 2002.
10. John McCarthy. The Robot And The Baby. *http://www-formal.stanford.edu/jmc*, 2002.
11. Marvin Minsky. *The Society of Mind*. Simon and Schuster, 1986.
12. R. Plutchik. *The Psychology and Biology of Emotion*. New York: Harper-Collins, 1994.
13. S. Reilly. Believable Social and Emotional Agents. Technical Report CMU-CS-96-138, School of Computer Science, Carnegie Mellon University, 1996.
14. Ira J. Roseman, Martin S. Spindel, and Paul E. Jose. Appraisals of Emotion-Eliciting Events: Testing a Theory of Discrete Emotions. *Journal of Personality and Social Psychology*, 59(5):899–915, 1990.
15. Juan D. Velásquez. Cathexis: A Computational Model for the Generation of Emotions and their Influences in the Behavior of Autonomous Agents. Master's thesis, Massachusetts Institute of Technology, 1996.
16. Fabio Zambetta, Graziano Catucci, and Fabio Abbattista. Cindy: A 3D Virtual Entertainer. *AISB Journal*, 1(3):291–302, 2003.

Intelligent Forecast with Dimension Reduction

István Juhos[1] and György Szarvas[1]

[1] Department of Computer Algorithms and Artificial Intelligence, University of Szeged and
[2] Research Group on Artificial Intelligence, Hungarian Academy of Science

Summary. Time-series prediction can be interpreted in a way that is suitable for artificial intelligence learning. Two effective learning methods, Artificial Neural Networks and Support Vector Machines, are used to provide accurate non-linear models of the problem. In spite of the effectiveness of these methods we have to solve two problems. Firstly, time-series can have noise and a high dimensional embedding space. Secondly, the learning depends on several hyper-parameters that need to be set properly. To handle these problems we apply noise and dimension reduction techniques and model selection to get suitable hyper-parameters. Then, we introduce a meta-heuristic to refine the predictions of the selected models. Our experiments show improvements in the quality of predictions of a real-life problem compared to two 'benchmark' algorithms.

Key words : time-series prediction , model selection, principal component analysis, neural networks, support vector machines

1 Introduction

Time-series prediction is important in a wide range of areas and has numerous applications. Take, for instance, forecasting traffic or the weather, predicting product demand in business or forecasting share prices in financial markets. Since estimates about future changes and developments are important for making decisions and taking actions, there is a constant need for more precise forecasting techniques. The aim of this paper is to describe a novel learning framework that offers a solution to the problem of forecasting dealing with the external influences. The new aspects of this work combine various artificial intelligence (AI) methods for tackling such problems, i.e. unsupervised learning for noise and dimension reduction; supervised learning for making models of the problem; a heuristic search for model selection and a meta-heuristic for merging the models of the different learning techniques to obtain better results. The supervised learners we decided to use are the Multi Layer Perceptron (MLP) and Support Vector Regression (SVR) techniques as it is known they have good approximation abilities [3, 6] needed for time-series prediction. There are several applications of them in this field, e.g. in [13, 7, 17, 12]. However, the new promising support vector techniques seem to be better than a similar neural network one because of its more sophisticated learning approaches, but in some cases neural networks can perform better and are used more frequently in practice (see [10]), so we have decided to use both. Automatic model selection is crucial for making the system useful in practice. The main aim is to find the best fitting model from a range of models (possibly infinite), which means the right choice of values for its hyper-parameters. Of course these parameter values must be obtained in a reasonable time. Several model selection techniques were introduced for MLP and SVR learners (see [19, 1, 15, 2]). There are many validation-based heuristic search methods among them owing to the large number of parameters. Two heuristic-based optimizers are commonly used for these kind of problems: the Evolutionary Algorithm [11] and Simulated Annealing (SA) [9]. We chose SA because of its lower resource-consuming characteristics. The meta-heuristic examines the models provided by the two learning methods after the model selections, then makes suggestions about which one should be used in forecasting. Time-series data usually comes from real-life situations with noise and a high embedding dimension, both of which can wreck the predictions. A reduction of these undesirable components can be achieved by applying linear Principal Component Analysis (PCA) [8], which is optimal in the mean square sense and with Kernel Principal Component Analysis [18], the corresponding non-linear version of PCA. Both of them achieve noise reduction and dimension reduction by extracting the features considered important by them. Using the above artificial intelligence methods we will construct a comprehensive, robust prediction framework. The paper is organized as follows. In the next section we provide an overview of time-series prediction. Then we introduce the things necessary for our prediction framework: learning, model selection, dimension reduction and the meta-heuristic, described in Sections 3,4,5 and 6 respectively. In Section 7 we assemble them in a framework. The experiments we performed are then discussed in Section 8. Lastly, in Section 9, we summarize our results and offer suggestions for future work.

2 Time-series Prediction

In time-series prediction problems, the aim is to predict in advance the value of a variable that varies in time using previous values and/or other variables. Typically the variable is continuous, so time-series prediction is usually a specialized form of regression. Several authors (e.g., [14]) transformed the temporal dimension of a time-series into a spatial vector of the l dimension embedding space by taking a moving window over the last l elements of the series. We can divide it into two kinds of forecast. First, when we do not use any other information apart from the time-series being examined (that is, predicting without external factors). Second, predicting with external factors when some external information is available.

Prediction without external factors

Suppose we have a real-valued time-series $\{y_t\}_{t=1}^n$, the historical values of the series. When other time-series which can affect the y-series are not given, the task is to predict y_{n+k} values with $k > 0$ based on the historical values. In other words

$$y_{n+k} = h(y_n, y_{n-1}, \dots, y_{n-(l-1)}), l > 0 \tag{1}$$

The usual way to make the prediction is to find an appropriate l and a function which describes the relation between l consecutive elements and the next element of the series. Here l denotes the window size and k the horizon.

Prediction with external factors

In some cases other time-series are known which can influence the y-series being examined. They are called the external factors. If several factors are available we can represent them as a vector-series consisting of more than one scalar time-series, that is $\{\mathbf{z}_t\}_{t=1}^n = \{z_{t1}, \dots, z_{tm}\}_{t=1}^n, m > 0$. We can include this information in the 1.

$$y_{n+k} = h(\hat{\mathbf{z}}_{n+k}, y_n, y_{n-1}, \dots, y_{n-(l-1)}), \quad k, l > 0 \tag{2}$$

Now suppose that the $(n+k)^{th}$ value of the $\mathbf{z}$-series or its estimate is known from some source, which will be denoted by $\hat{\mathbf{z}}_n$. Unfortunately, the problem of obtaining information about $\hat{\mathbf{z}}_n$ is similar to that for 1, hence the $\mathbf{z}$-series prediction should be made before y_n prediction. When the aim is to predict only the future value of y_{n+k}, the prediction is called a one-step-ahead prediction. However, if we intend to estimate values beyond y_{n+k}, we have to use the previously predicted values in the h function and we call this a many-step-ahead prediction, especially with N-step-ahead predictions where the intention is to forecast the next N values.

3 Inductive Learning

The inductive learning of a concept requires recognizing a hypothesis for this concept after presenting training attributes and supervising it by defining the classification. During the learning/training process these are generally given in the following format:

$$\underbrace{\underbrace{x_{i1}, x_{i2}, \dots, x_{il}}_{attributes}, \underbrace{y_i}_{class},}_{i^{th} instance} \quad i \in \mathbb{N} \tag{3}$$

In order to seek a relation (the hypothesis) between the attributes and their classes, a h function based on the training instances has to be approximated:

$$y_i = h(x_{i1}, x_{i2}, \dots, x_{il}) \tag{4}$$

The above formula is suitable for the prediction schemes in Section 2 if we replace the arguments of the h function. The attributes of the i^{th} instance mean the i^{th} data window while y_i is the value we need to predict. From here on we will make use of this more general notation. The AI learners we applied were Multi Layer Perceptron and Support Vector Regression, both of them have good approximation characteristics [3, 6].

3.1 Multi Layer Perceptron (MLP)

The two-layered MLP is capable of approximating arbitrary finite sets of real numbers [3]. Hence, we used a maximum two hidden layered MLP with sigmoid activation function ($\theta(\mathbf{x}) = 1/(1 + e^{-\mathbf{x}})$), where the input and output layers have linear units. When the l attributes of the i^{th} learning instance takes the form $x_{i1}, \ldots, x_{il}$ then the MLP approximation is

$$y_i \approx \sum_{r=1}^{l^2} (w_r^{out} \theta(\sum_{s=1}^{l^1} w_s^{2r} \theta(\sum_{t=1}^{l} w_t^{1s} x_{it} + w_{bias}^{1s}) + w_{bias}^{2r}) + w_{bias}^{out}) \tag{5}$$

$\theta(.)$: activation function; l^1, l^2: number of perceptrons in the 1^{st} and 2^{nd} layers; w_t^{1s}, w_m^{2r}, w_r^{out}: weights in the 1^{st} and 2^{nd} hidden and output layers; w_{bias}^{1s}, w_{bias}^{2r}, w_{bias}^{out}: biases in the 1^{st} and 2^{nd} hidden and output layers

Changing the weights is the basis of the learning process. The well-known backpropagation method with momentum is used for adjusting the weights during the training process[3]. Backpropagated MLP learning depends on the following parameters that need to be tuned: the number of neurons in the hidden layers, learning rate, momentum and number of training epochs.

3.2 Support Vector Regression (SVR)

There are two commonly used Support Vector Machines for regression, namely the ϵ-SVR algorithm and its extension the ν-SVR algorithm (see [16]). We chose the ν-SVR because it has an advantage, in contrast with ϵ-SVR, of being able to automatically adjust the width of the ϵ-tube around the function being approximated. An SVR maps the $\mathbf{x_i} = (x_{i1}, \ldots, x_{il})$ attributes to a generally higher dimension space, called the feature space via a $\phi : R^l \to R^L$, $L \geq l$ map function. Then it makes a linear fit according to (6) to a certain precision by optimizing the weights $\mathbf{w} = (w_1, \ldots, w_L)$ and w_{bias}:

$$y_i \approx \sum_{j=1}^{L} w_j \phi(x_{ij}) + w_{bias} \quad (= \langle \mathbf{w}, \phi(\mathbf{x}_i) \rangle + w_{bias}) \tag{6}$$

We can reformulate (6) by expanding the weight vector from the attribute vectors as a linear combination ($\mathbf{w} = \sum_{t=1}^{n} (\alpha_t^* - \alpha_t) \phi(\mathbf{x}_t), \quad \alpha_t^*, \alpha_t \geq 0$):

$$\sum_{t=1}^{n} (\alpha_t^* - \alpha_t) \langle \phi(\mathbf{x_t}), \phi(\mathbf{x_i}) \rangle + w_{bias} \quad (= \sum_{t=1}^{n} (\alpha_t^* - \alpha_t) \kappa(x_t, x_i) + w_{bias}) \tag{7}$$

where κ is a kernel function belonging to the ϕ mapping. To obtain α_t^*,α_t ν-SVR[4] maximizes the following quadratic problem for $C, \nu > 0$:

$$-\tfrac{1}{2} \sum_{i,j=1}^{n} (\alpha_i^* - \alpha_i) \kappa(x_i, x_j) (\alpha_j^* - \alpha_j) + \sum_{i=1}^{n} (\alpha_i^* - \alpha_i) y_i$$

subject to the constraints

$$\textstyle\sum_{i=1}^{n} (\alpha_i^* - \alpha_i) = 0; \quad \sum_{i=1}^{n} (\alpha_i^* + \alpha_i) \leq Cn\nu; \quad 0 \leq \alpha_i^*, \alpha_i \leq C \tag{8}$$

[3] Available in the Weka library [20]

[4] Available in the LibSVM library [2]

A solution of the above equation provides coefficients for each $\mathbf{x}_i$ attribute vector in order to combine them to the $\mathbf{w}$ weight vector. These coefficients suggest the relevance of the attribute vectors. However, these values characterize well the training attributes in the ϕ mapped image space, our experiments confirm that: knowing the coefficients in the unknown image space does not result direct analysis of the data in the original space. We used the radial basis function ($\kappa(\mathbf{x_i}, \mathbf{x_j}) = e^{-\gamma\|\mathbf{x_i}-\mathbf{x_j}\|^2}$) as a kernel function because it is frequently employed in function estimation problems [4]. To get a better performance some parameters in the ν-SVR algorithm still have to be tuned: the kernel parameter γ and the regularization parameters C, ν.

4 Model Selection

Machine learning techniques suffer from certain problems like overfitting. These problems occur because of improperly chosen learning hyper-parameters, e.g. the number of neurons for a neural network. In addition, the prediction tasks can require some other hyper-parameters like the window size in the windowing mentioned in Section 2. The choice of these parameters can lead to a better or worse forecast as well. Hence, the prediction needs good settings that define a good model with the underlying methods. Finding good parameters is hard because of the induced search space and a function which measures their goodness in general has unknown or bad behaviour from an optimization point of view (such as differentiability problems and the existence of many local optima). Our model selection is based on a validation process, where historical data is divided into training and test sets. A learner uses a training set for making a hypothesis and we can validate this on the test set by comparing the hypothesis-provided estimation with the expected real values. A comparison of the two gives us a measure of the goodness of fit. Based on this measure the model selector can decide whether to accept or modify the parameters and repeat this process in the hope of getting a better solution. Our goodness/fitness measure (the f function of a $\mathbf{p}$ parameter vector) is the Normalized Root Mean Squared Error (NRMSE) (9). A normalized error function is appropriate to compare methods on different kinds of data.

$$f(\mathbf{p}) = \sqrt{\sum_{i=1}^{n} \frac{(y_i - \hat{y}_i(\mathbf{p}))^2}{n\sigma^2}}, \tag{9}$$

where σ is the standard deviation of the y-series. Essentially, the model selection task is to minimize the fitness function over the vector space induced by the parameters. A heuristic method is used for optimization because of above-mentioned bad behaviour of the problem. This is Simulated Annealing (SA) [9], which is a robust optimizer. SA can avoid getting stuck in a local optima and does not demand differentiability.

4.1 Simulated Annealing

Simulated Annealing (SA) is a heuristic method for locating the extrema of a function. In our case, we are looking for the minimum of the fitness function

f. The algorithm has a physical motivation. During the annealing of liquids, the fine structure of the material follows the principle of optimal disposition. The pseudo-code of the applied SA method is found below:

Procedure Simulated Annealing

```
  $t$ = initial temperature; $\mathbf{p}_{best} = \mathbf{p}$ = a random parameter vector
  while(stop condition is false)
      $\mathbf{p}' = \mathbf{p} + \mathbf{u}D_t$; $\Delta f = f(\mathbf{p}') - f(\mathbf{p}_{best})$
      if $(\Delta f < 0)$ $\mathbf{p}_{best} = \mathbf{p}'$ else $\mathbf{p}_{best} = \mathbf{p}'$ with probability $e^{-\Delta f/t}$
      $t = \xi t$; update $D_t$
  end while,
```

where t is the temperature which decreases with annealing rate ξ; u is a random number vector whose elements lie in the range {-1,1} and D_t is a diagonal matrix that defines the maximum changes allowed in each variable. Maximum changes decrease in direct proportion to t. The algorithm employs a random search that not only accepts changes that decrease the objective function f, but also some changes that increase it with probability $e^{-\Delta f/t}$. This can prevent it from becoming trapped at a local optimum. Here we optimize the tunable, learner-specific parameters mentioned in Section 3. However, several other parameters could also be tuned, but these play a relatively minor role in the prediction procedure.

5 Dimension Reduction

Handling the high dimensional embedding space with too few past instances can cause another problem for the learning, namely one that is known as the curse of dimensionality [5]. In addition, there is usually noise in the data. A good way of dealing with it can be a dimension reduction of the dimension of the data from l to Γ $(\Gamma < l)$. Two kinds of reduction schemes are employed: a linear and a non-linear one. From here on we will assume that the training attributes are centered $(\sum_{i=1}^{n} x_i = 0)$ as this is needed for the dimension reduction algorithms discussed below.

5.1 Linear Dimension Reduction with Principal Component Analysis (PCA)

PCA is an optimal linear dimension reduction technique in the mean-square sense [8]. After dimension reduction the learning requires less computational effort and the data usually has reduced noise and is free of the curse of dimensionality. The basic idea in PCA is to find the components $s_{i1}, s_{i2}, ..., s_{i\Gamma}$, $i \in \{1, \ldots, n\}$ so that they have the maximum amount of variance with Γ linearly transformed components. PCA can be defined in an intuitive way using a recursive formulation. Now let us define the direction of the first principal component by

$$\mathbf{v}_1 = \arg \max_{\|\mathbf{v}\|=1} E\{|\langle \mathbf{x}_i, \mathbf{v} \rangle|\}_{i=1}^{n}, \quad \mathbf{v} \in \mathbb{R}^l \tag{10}$$

where $\mathbf{x}_i$-s are the population of the training attributes and E denotes the mean. Thus the first principal component is the projection in the direction in which the variance is maximized. Having determined the first $k-1$ $(1 < k < l)$ principal components, the k^{th} principal component is found by calculating the principal component of the residual:

$$\mathbf{v}_k = \arg\max_{\|\mathbf{v}\|=1} E\{|\langle \mathbf{v}, (\mathbf{x}_i - \sum_{j=1}^{k-1} \mathbf{v}_j \langle \mathbf{x}_i, \mathbf{v}_j \rangle)\rangle|\}_{i=1}^{n}, \quad \mathbf{v} \in \mathbb{R}^l \tag{11}$$

The principal components are then given by $s_{ij} = \langle \mathbf{x}_i, \mathbf{v}_j \rangle$. We have l components of the data, but many of them usually have a small variance. Depending on the variance, we keep the first Γ components which have a significant variance. In practice, the computation of $\mathbf{v}_i$ can be accomplished using the covariance matrix $C = E\{\mathbf{x}_i {\mathbf{x}_i}^T\}_{i=1}^{n}$ of the training attributes, assuming that $\mathbf{x}_i$-s are column vectors. The $\mathbf{v}_i$ are the eigenvectors of C that correspond to the n largest eigenvalues of C.

5.2 Non-linear Dimension Reduction with Kernel Principal Component Analysis (KPCA)

By non-linearizing the eigen-problems of the PCA method (12) we can turn it into the KPCA method (see [18]). It usually non-linearly maps the training attributes to a higher dimension space where the vectors have more discriminant features. Applying the PCA method in the higher dimension space, we can extract better features from the training attributes. However, as we have more principal components than the dimension of the original space, we keep as many as necessary for the reduction.

$$C\mathbf{v} = \frac{1}{n}\left(\sum_{i=1}^{n} \mathbf{x}_i {\mathbf{x}_i}^T\right)\mathbf{v} = \lambda \mathbf{v}, \quad \lambda \in \mathbb{R}, \mathbf{x}_i, \mathbf{v} \in \mathbb{R}^l \tag{12}$$

Eigenvectors with $\lambda \neq 0$ lie in the $Span\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, thus each of them can be reproduced by a linear combination of the training attributes. Rewriting 12 using this, we get:

$$\mathbf{v} = \sum_{i=1}^{n}\left(\frac{1}{\lambda n}\langle \mathbf{x}_i, \mathbf{v}\rangle\right)\mathbf{x}_i \quad (= \sum_{t=1}^{n} \alpha_i \mathbf{x}_i, \quad \alpha_i \in \mathbb{R}) \tag{13}$$

Making use of these two equations (12, 13) and mapping vectors from the l dimension space to a higher $\mathcal{L}$ dimension space by the $\Phi : R^l \to R^{\mathcal{L}}$, $\mathcal{L} \geq l$ map function, leads to the following equation:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle \alpha_j \Phi(\mathbf{x}_i) = n\lambda \sum_{j=1}^{n} \alpha_j \Phi(\mathbf{x}_j), \quad \lambda \in \mathbb{R}, \Phi(\mathbf{x}_i) \in \mathbb{R}^{\mathcal{L}} \tag{14}$$

Rewriting 14 as an eigen-problem, we get the following:

$$K\boldsymbol{\alpha} = \lambda \boldsymbol{\alpha} \tag{15}$$

$$\boldsymbol{\lambda} \in \mathbb{R}, \quad \alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n), \quad K(i,j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle = \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

K is an $\mathcal{L} \times \mathcal{L}$ matrix called the kernel matrix. By choosing an appropriate Φ function we do not have to compute the explicit mapping of the vectors each time; we only need to compute kernel matrix once using the kernel function κ. Employing the terminology of Section 5.1 the kernel principal components of an $\mathbf{x}_j$ attribute is given by $s_i = \langle \mathbf{V}_i, \Phi(\mathbf{x}_j) \rangle = \sum_{i=1}^{n} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}_j)$, where $\mathbf{V}_i$ is the i^{th} eigenvector in the $\mathcal{L}$ dimension space. Here, the commonly used radial basis function ($\kappa(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x_i} - \mathbf{x_j}\|^2}$) was the one we chose as the kernel function.

6 Meta-Heuristic

After the model selection process the learners supply hypotheses. Then we introduce a meta-heuristic method to refine the hypotheses to a meta-hypothesis. Such a meta-hypothesis proves useful here as it brings significant improvements in experimental results. Now let us consider a set of learnt hypotheses $H = \{h_1(.), h_2(.), \ldots, h_r(.)\}$ where one of them will be chosen to predict a future value. These choices can be represented by a decision function.

Fitness-Based Meta-Heuristic (FMH)

With this decision function, we choose a hypothesis $h_i(.)$ from the set H which has the best fitness value on the test set. It means we choose the one which has the smallest empirical error on the unseen test data. Hence we can expect better results in the future than those provided by the original hypotheses.

7 Time-Series Prediction Framework

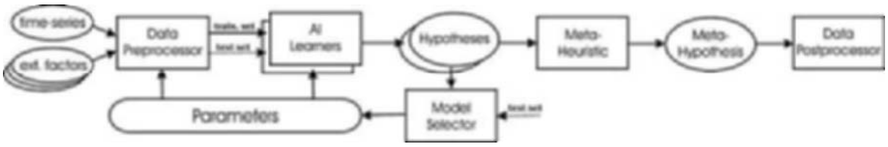


Fig. 1. Time-Series Prediction Framework

Consider the framework represented in Figure 1. First, the Data Preprocessor receives, as input, the historical values of the time-series being examined and, optionally, external factors. The Preprocessor performs windowing in the way described in Section 2, then normalizes attributes and the class of instances. Normalization can be achieved by scaling the data so that it lies in a [0,1] or [-1,1] interval or translating it so that it has a zero mean and unit variance. Afterwards, noise reduction and dimension reduction is performed. Finally, the Preprocessor separates the input data into a training set (the first, bigger part) and a test or validation set (the last, smaller part). The prediction interval is the continuation of the input sequence, so splitting the instances preceding the forecast interval for validation purposes seems to be a good choice. Note that there are free hyper-parameters to choose here in the preprocessing phase. We have to choose the appropriate window size, the normalization type, the ratio of the dimension reduction and the size of the training and test set. Sometimes these values can be determined by analyzing the data, sometimes they cannot. In the latter case the Model Selector has to provide appropriate values for them. In our experiments we fixed the parameter values by analyzing the behaviour of the data. This led to a smaller, but still high-dimensional space of the remaining hyper-parameters. After the preprocessing stage, the AI Learners try to devise hypotheses based on the training set with their initial Parameters. Then, the Model Selector verifies these hypotheses on the test set and decides how to modify the Parameters in order to get better hypotheses. The Model Selector iteratively refines the parameters then retains the settings found best during the iterations. The Meta-Heuristic tries to make a Meta-Hypothesis that is a decision function between the Learners at each prediction point. The decision is obtained by measuring the performance of the hypothesis of the Learners on the test set using the best settings found in the model selection phase. After creating such a decision function we retrain the Learners on the whole historical dataset (training+test sets), since, using the values closest to the prediction interval (i.e. using the test set) in the training gives more precise learner-specific hypotheses. A better prediction accuracy can be achieved in this way. The prediction is made via the Meta-Hypothesis, which decides which refined hypotheses should be used at each prediction point. The forecast is normalized because the data was preprocessed, so the Postprocessor denormalizes it to get the final result.

8 Experiments

In our experiments real meteorological data was used, which came from a monitoring station located in the city of Szeged in Hungary. The time-series examined was Nitric Dioxide (NO_2) and consisted of a six-month dataset from the period between 1st September in 2000 - 23th February in 2001 in 1-hour averages. One-step-ahead prediction was applied for a two-day forecast (22th and 23th February) from the period 1st September - 21rd February. For demonstration purposes we used the real values of the Nitric Oxide (NO) time-series as the external factor of the NO_2-series. A strong connection between the two series can be seen in [13]. The NO_2 time-series has natural 24-hour periods, confirmed by the autocorrelation diagram in Figure 2. Here we notice peaks at every 24 hours.

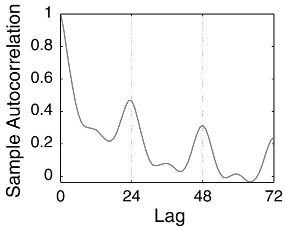
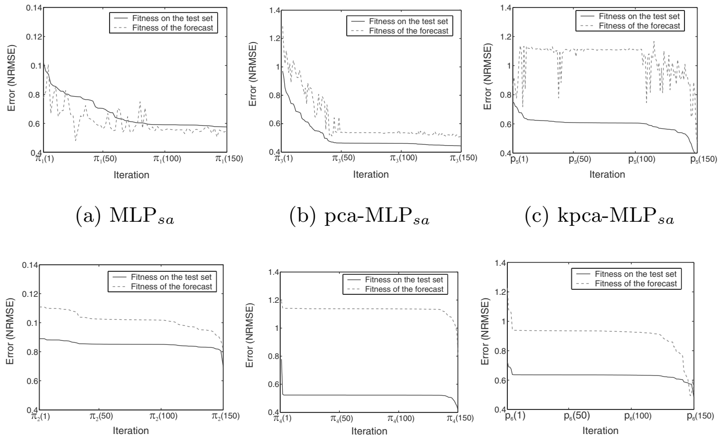


Fig. 2. Autocorrelation of the NO_2-series

It is expedient to choose 24 for the embedding dimension, i.e. the window size. For normalization we chose [0,1] scaling for the MLP learner and normalized the data so that it had a zero mean and unit variance for the SVR learner. These transformations seemed to be the most appropriate in some preliminary tests. In fact normalization is necessary and in practice there was not much difference between the normalization techniques mentioned in Section 7. An MLP and a ν-SVR were trained for each hour of a day by model selection with and without dimension reduction to see its effect. A MLP and a ν-SVR learner were each trained for each hour (i.e. 24×2 learners). Owing to the stochastic behaviour of the heuristic model selection we took averages of the predictions of three independent runs. The averages provide two learner-specific hypotheses, which will be denoted by MLP_{sa} and $\nu\text{-SVR}_{sa}$ when dimension reduction was not done. Using these hypotheses the FMH meta-heuristic yielded further results based on the decision of the independent runs. The previous notations were extended with 'pca-' and 'kpca-' prefixes when PCA or KPCA was applied. We chose the standard parameters of the learners defined by the libraries used [20, 2] as a starting point of the model selection, then the Simulated Annealing algorithm evolved them in 150 iterations with a $\xi = 96\%$ exponential annealing schedule from initial temperature 4. We retained the principal components of the PCA with 95% cumulative variance. In the case of KPCA we had more principal components than the number of attributes and the prediction process was more sensitive to the remaining number of components. Hence we selected those components with the largest variances for which the prediction error based on the test set (i.e., the validation set of the model selection) was minimal and gave a dimension reduction. The gamma parameter of KPCA was set to 0.01 so it would have sufficient variance. The division ratio of the historical data was 80/20% for training and test sets in the model selection phase to avoid overfitting. Figure 3 shows the behaviour of the forecast error in proportion to the test error, i.e. fitness during the model selection. The iterations were reordered in decreasing error values on the test set to see the correlation. The figures asymptotically show that the smaller the fitness, the smaller the prediction error. Furthermore, we can see that the ν-SVR result is smoother because of the more robust learning scheme, while MLP forecast errors jumped about during iterations.

Dimension reduction provided a more regular learning scheme with higher accuracy (see Figures 3(b), 3(c), 3(e), 3(f) and Table 8). As can be seen, both of their forecast errors converge to the minimum in direct proportion to the test error in the model selection process. It suggests that the model selection process works well, and improves the efficiency of the hypotheses. We compared the performance of our framework with two reference algorithms commonly used for this type of problem. The first reference algorithm is called Persistence and models the persistence of the values of the days. A prediction value of a future hour has the same value as that for the same hour of the previous day. It cannot handle external factors and there is no point in using dimension reduction. This simple technique works well on this problem because the series has a 24-hour periodicity. The second reference algorithm is called Linear Regression (LR). It was shown in [13] that both work well on this type of problem and outperform some MLP results in certain cases. Linear Regression without the *NO* external factor yields a 0.64 prediction



(a) MLP$_{sa}$ (b) pca-MLP$_{sa}$ (c) kpca-MLP$_{sa}$

(d) ν-SVR$_{sa}$ (e) pca-ν-SVR$_{sa}$ (f) kpca-ν-SVR$_{sa}$

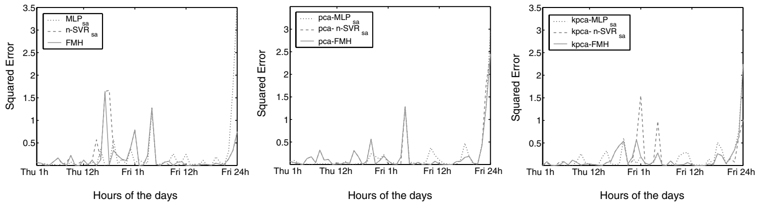
Fig. 3. The relation between the test and prediction errors during model selection steps for certain hours. The corresponding results were reordered using certain permutations $\pi_{\{1,2,3,4,5,6\}}$ to get decreasing values.

error (NRMSE) for the NO_2-series on the forecast interval in question. However, when the NO external factor is included the prediction error drops to 0.47. We can thus see the effect of this external influence on the prediction process. This shows the importance of the inclusion of external factors in the framework. We use the NO supported prediction of the NO_2-series according to Section 2. The forecast errors of MLP and ν-SVR were 0.72 and 0.52 using the libraries' standard parameter settings without any model selection. With PCA these the corresponding values were 0.54 and 0.48, and with KPCA we obtained values of 0.48 and 0.68 respectively. It seems that the learners with standard settings cannot compete with the parameterless LR method (which produced a value of 0.47), even after the improvements brought about by dimension reduction. By inserting the MLP and the ν-SVR learners into our model selection framework, we can outperform the reference methods (see Table 8). While MLP$_{sa}$ and ν-SVR$_{sa}$ provided by the model selection barely competes with the reference ones, FMH works with great success and we can see improvements of up to 13.24% over the best reference algorithm. Using PCA leads to more uniform and reliable predictions, but the performance of the meta-heuristic is more modest (Table 8). The greater stability is implied in Figure 4. KPCA provides very good results when combined with the kernel-based ν-SVR$_{sa}$ and provides good results with MLP$_{sa}$ as well. Overall, FMH yields the best results here with the improvements of 14.52% over the best benchmark algorithm.

		Improve.			Improve.			Improve.
Method	Error	Best ref.(LR)	Method	Error	Best ref.(LR)	Method	Error	Best ref.(LR)
LR	**0.47**	-	pca-LR	0.61	-	kpca-LR	0.69	-
Persist.	0.79	-	Persistence	0.79	-	Persistence	0.79	-
ANN_{sa}	0.47	**0.38%**	pca-ANN$_{sa}$	0.43	**7.17%**	kpca-ANN$_{sa}$	0.41	**12.26 %**
ν-SVR$_{sa}$	0.46	**2.29%**	pca-ν-SVR$_{sa}$	0.44	**5.93%**	kpca-ν-SVR$_{sa}$	0.41	**12.87 %**
FMH	0.41	**13.24%**	pca-FMH	0.43	**7.63%**	kpca-FMH	0.4	**14.52 %**

Table 1. Errors (NRMSE) and improvements of the reference methods of the NO_2-series predictions with an NO external factor. The left table shows the results without dimension reduction, while the middle and right ones respectively show the results with PCA and KPCA dimension reduction.

In Figure 4(a) we can see the errors in the forecast given by the MLP$_{sa}$, ν-SVR$_{sa}$ algorithms and the meta-heuristic. Figures 4(b) and 4(c) show the corresponding values using PCA and KPCA. The figures reveal the refining effect of the meta-heuristic, e.g. at 'Fri 12h' in Figures 4(a), 4(b) and 4(c). We observed unpredictable behaviour of the original time-series, e.g. at 'Fri 24h', where larger prediction error could be observable. There should have been some other external influence which was not considered in our experiment. Even though the undesired behaviour of the series some model could provide good estimation.



(a) without PCA (b) with PCA (c) with KPCA

Fig. 4. Squared Errors of the predictions in the two forecasted days divided by the square of the std. deviation of the series to obtain comparable normalized values.

9 Conclusions

As the experiments clearly show, the applied forecasting framework can perform well on a real-life problem. The forecasting of meteorological data is hard because the concentrations of some atmospheric components fluctuate wildly and depend on several factors. In many cases the two reference algorithms, Persistence and Linear Regression prove successful at forecasting the future values of a time-series [13]. Moreover, our findings demonstrate that the model selection and the applied meta-heuristic can bring about significant improvements in the effectiveness of the learning methods used. Using

Simulated Annealing to get better accuracies may outweigh the higher computation time required for it. Its practical usability and the fact that it lends itself to automation, are also good reasons for using it. Introducing dimension reduction techniques into the framework might lead to more reliable and robust learning, and a better performance from an accuracy point of view. The applied algorithms are good representatives of the linear and nonlinear methods, but there are many other approaches to transform raw data according to eliminate some unnecessary components. Trying other algorithms may form the field of future investigation. Fortunately, the framework has modular structure that also allows us to use other AI techniques and tune additional hyper-parameters. These possibilities suggest ways in which our framework might be extended. Without a doubt the application of Machine Learning techniques to problems like the above can be relatively simple and worth using.

References

1. Castillo P A, Merelo J J, González J, Rivas V, Romero G (1999) SA-Prop: Optimization of Multilayer Perceptron Parameters Using Simulated Annealing. In: Touretzky D, Mozer M, Hasselmo M (eds) Proceedings of 5th International Work-Conference on Artificial and Natural Neural Networks 1:661–670
2. Chang C-C, Lin C-J (2001) LIBSVM: a Library for Support Vector Machines (Version 2.31). http://citeseer.ist.psu.edu/chang01libsvm.html
3. Chester D (1990) Why Two Hidden Layers are Better Than One. In: Proceedings of International Joint Conference on Neural Networks '90 265–268
4. Evgeniou T, Pontil M, Poggio T (2000) Regularization Networks and Support Vector Machines. Advances in Computational Mathematics 13(1):1–50
5. Friedman J H (1994), An overview of predictive learning and function approximation. In: Cherkassky V, Friedman J H, Wechsler H (eds) From Statistics to Neural Networks, Theory and Pattern Recognition Applications 1–61
6. Hammer B, Gersmann K (2003) A Note on the Universal Approximation Capability of Support Vector Machines. Neural Processing Letters 17(1):43–53
7. Gardner M V, Dorling S R (1998) Artificial Neural Networks (the Multilayer Perceptron). Atmospheric Environment 32(14/15):2627–2636
8. Jolliffe I T (1986) Principal Component Analysis. Springer-Verlag, New York.
9. Kirkpatrick S, Gerlatt C D Jr, Vecchi M P (1983) Optimization by Simulated Annealing. Science 220(4598):671–680
10. KDnuggets (2004) Polls : Data mining techniques (Nov 2003). http://www.kdnuggets.com/polls/2003/data_mining_techniques.htm
11. Michalewicz Z (1992) Genetic Algorithms+Data Structures=Evolution Programs. Springer, Berlin
12. Müller K R, Smola A J, Rätsch G, Schölkopf B, Kohlmorgen J, Vapnik V (1997) Predicting Time Series with Support Vector Machines. In: Gerstner W (eds) Proceedings of 7th International Conference on Artificial Neural Networks 999–1004
13. Perez P, Trier A (2001) Prediction of NO and NO2 concentrations near a street with heavy traffic in Santiago, Chile. Atmospheric Environment 35:1783

14. Refenes A N (1993) Constructive Learning and Its Application to Currency Exchange Rate Forecasting. Neural Network Applications in Investment and Finance Services, Probes Publishing, Chicago
15. Rocha M, Cortez P, Neves J (2003) Evolutionary Neural Network Learning. In: Callaos N, da Silva I N, Molero J (eds) Proceedings of 11th Portuguese Conference on Artificial Intelligence 24–28
16. Schölkopf B, Bartlett P, Smola A, Williamson R (1998) Support Vector Regression with Automatic Accuracy Control. In: Niklasson L, Bodén M, Ziemke T (eds) Proceedings of 8th International Conference on Artificial Neural Networks 111–116
17. Schölkopf B, Burges C J C (1998) Advances in Kernel Methods Support Vector Learning, The MIT Press, Cambridge
18. Schölkopf B,Smola A, Müller K-R (1998) Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation 10:1299–1319
19. Yao X, Liu Y (1997) A New Evolutionary System for Evolving Artificial Neural Networks, IEEE Transactions on Neural Networks 8(3) 694–713
20. Witten I H, Frank E (2000) Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco

Stochastic Algorithm Computational Complexity Comparison on Test Functions

Nicola Cesario[♯], Palma Petti[§], Francesco Pirozzi[♯]

[♯]SST Corporate R&D STMicroelectronics
via Remo de Feo, 1, Arzano(NA), 80022, Italy
nicola.cesario@st.com
francesco.pirozzi@st.com
[§]IIASS Istituto Internazionale per gli Alti Studi Scientifici
Vietri sul mare(SA), Italy

Key words: Evolutionary algorithms, evolution strategies, differential evolution, particle swarm optimization.

1 Introduction

The *Evolutionary Algorithms*(EA), see [1] and [2], are stochastic techniques able to find the *optimal solution* to a given problem. The concept of *optimal solution* depends on the specific application, it could be the search of the global minimum of a complicated function. These algorithms are based on *Darwin* theories about *natural selection*. Natural selection allows to survive only best individuals (that is individuals more suitable to fit environment changes); in this way there is a generalized improvement of the entire population. Only the most performing individuals can transfer their genotype to the descendants.
In the EA the parameter measuring individuals performance (in literature known as individuals *fitness*) is called *fitness function*. Time goes on by discrete steps. Starting by an initial population randomly generated, the process of evolution takes place. The most used operators that allow to obtain the new generation are: *Reproduction*, *Recombination*, *Mutation* and *Selection*. Let's to consider more formally these statements. Given a generic fitness function F defined in a N-dimensional parameters space, Y, and with values in an M-dimensional space Z:

$$F : \mathbf{y} \in \mathcal{Y} \to F(\mathbf{y}) \in \mathcal{Z} \tag{1}$$

an optimization problem can be formalized as follow: *to find the parameters vector* $\hat{\boldsymbol{y}} \in \mathcal{Y}$ in which the function has the *optimal* value referred to the problem we are solving

$$F(\hat{\mathbf{y}}) := \text{opt} \;\; F(\mathbf{y}) \tag{2}$$

where

$$\mathbf{y} = (y_1, \ldots, y_N) \qquad \hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_N) \tag{3}$$

The components y_i of $\mathbf{y}$ are called *object variables.* EA operate on a population B of individuals each of that represents a point in the search space. In general the *Reproduction* selects one ore more parents to create, by *Recombination*, one or more descendants. In order to introduce an innovation in genotype *Mutation* is considered. After all these operations, *Selection* is applied to create the new population.

The paper is composed by an introductive section in which we shortly explain the peculiarity of all the considered algorithms. In the following we show the results of our experiment on some literature *hard* test functions. In conclusion, we propose a new hybrid algorithm with better performance than the others analyzed.

2 Brief on Test Case Stochastic Algorithms

In this section we point on description of the before mentioned operators for each algorithm considered in our analysis.

2.1 Evolutionary Strategy ES-(1+1)

The ES-(1+1) is a particular case of the more general ES-($\mu/\rho \overset{+}{,} \lambda$), where μ is the total number of individuals in the population, ρ is the number of individuals taking place to the reproduction phase, λ is the number of descendants. The symbols $\overset{+}{,}$ indicates the different set of individuals on which the selection operator must operate: in the *comma* selection only the population of descendants is considered in order to create the next population whereas in the *plus* selection the choose of best individuals is made considering both parents and descendants. Hence, in the ES-(1+1) we have an “asexual” reproduction and the choose of new individuals is made in the population of parents and descendants. This states that only the mutation operator is used to create descendants.

In literature, ES-(1+1) is used in several optimization problems because it is conceptually simpler than other algorithms and especially because it has a more satisfactory performance in the hypersphere model, see [1]. In figure 1 there is the ES-(1+1) pseudo-code.

In line 6 of figure 1 the descendant $\tilde{\mathbf{y}}$ is generated by the action of the mutation operator:

$$\tilde{\mathbf{y}} = \mathbf{y}_p + \mathbf{z} \qquad \text{with} \qquad \mathbf{z} = \sigma(\mathcal{N}(0,1), \mathcal{N}(0,1), \ldots, \mathcal{N}(0,1)) \tag{4}$$

where $N(0,1)$ is the normal distribution with zero mean and unitary variance and σ is the *mutation strength*, this parameter is initially fixed to 0.1.

```
$ES-(1+1)$-Algorithm
Begin
    $g = 0;$
    initialize$\Big( \mathcal{B}_{\mu}^{(0)} = \Big\{ \Big( \mathbf{y}_m^{(0)}, \sigma_m^{(0)} \Big) \mid m = 1,..,\mu \Big\} \Big);$
    Repeat
        $F_m = F(\mathbf{y}_m);$
        $\tilde{\mathbf{y}} = \mathbf{y}_m + \sigma(\mathcal{N}(0,1), \mathcal{N}(0,1), \ldots, \mathcal{N}(0,1));$
        $\tilde{F} = F(\tilde{\mathbf{y}});$
        If $\tilde{F} \leq F_m$ Then $\mathbf{y}_m = \tilde{\mathbf{y}};$
        $g = g + 1;$
     Until stop_criterion
    End
```

Fig. 1. ES-$(1+1)$ algorithm for fitness function minimization $F(\mathbf{y})$.

An innovation we have introduced is the using of a kind of *simulated annealing* (SA) to change the σ value. The idea is opposite to the *classical viewpoint* in searching minimum of a fitness function, in fact when we approaches a minimum instead of restricting the search radius we increase it in order to not fall in a local minimum. Figure 2 shows the SA pseudo-code. For a more detailed comprehension of SA pseudo-code we define the following parameters:

- $g_{back} \equiv$ generational step by which we go back to calculate Q (fixed)
- $Q \equiv$ success probability of descendants in the g_{back} interval
- $P \equiv$ annealing probability (fixed)
- $R^{(g)} \equiv$ current iteration hypersphere radius
- $R^{(g+1)} \equiv$ next iteration hypersphere radius
- $a \equiv$ amplification factor for R

2.2 Evolution Strategy Self-Adaptation ES-(1+5)-σSA

This ES algorithm differs by the ES-(1+1) either because λ is 5 or, that is the most important thing, because this algorithm is *Self-Adapting.* In order to understand what self-adapting means, we must give some definitions. We define *endogenous parameter* a parameter which could change during evolution and which is part of the genotype. In this case, there is only one endogenous parameter: σ. We, also, define *exogenous parameters* the ones that not change during evolution. In the ES-(1+5)-σSA, these are λ and μ.

In ES-(1+5)-σSA, σ is a part of the genotype of each individual and then it is subjected to recombination and mutation like the other parameters vector components. Figure 3 shows the ES-(1+5)-σSA pseudo-code.

In line 6 of figure 3 we can see the mutation operator for σ. In the continuous case the most famous mutation operator is the *log-normal* (see [1]) defined as

```
SA-Algorithm
Begin
    $g = 0$;
    If $g \geq g_{back}$
        For $i = g - g_{back}$ To $g$
            Success probability calculation: $Q$;
        End
    End
    If $Q \leq P$
            Restrict radius: $R^{(g+1)} = R^{(g)} * a$;
    else
            Enlarge radius: $R^{(g+1)} = R^{(g)}/a$;
    End
    $g = g + 1$;
End
```

Fig. 2. SA algorithm.

```
$ES - (1 + \lambda) - \sigma SA$-Algorithm
Begin
    $g = 0$;
    initialize$\left( \mathcal{B}_{\mu}^{(0)} = \left\{ \left( \mathbf{y}_m^{(0)}, \sigma_m^{(0)} \right) \mid m = 1, .., \mu \right\} \right)$;
    Repeat
        For $m$=1 To $\mu$
            For $l$=1 To $\lambda$
                $\tilde{\sigma}_{m,l} = \xi \sigma_m^{(g)}$;
                $\tilde{\mathbf{y}}_{m,l} = \mathbf{y}_m^{(g)} + \tilde{\sigma}_{m,l}(\mathcal{N}(0,1), \ldots, \mathcal{N}(0,1))$;
                $\tilde{F}_{m,l} = F(\tilde{\mathbf{y}}_{m,l})$;
            End
            $l_p$ =selection $\left( \tilde{F}_{m,1}, ..., \tilde{F}_{m,\lambda}, F_m \right)$;
            $\sigma_m^{(g+1)} = \tilde{\sigma}_{m,l_p}$;
            $\mathbf{y}_m^{(g+1)} = \tilde{\mathbf{y}}_{m,l_p}$;
            $g = g + 1$;
        End
    Until $stop\_criterion$
End
```

Fig. 3. $ES - (1 + \lambda)$ algorithm with σ-self-adaptation.

follow:

$$\tilde{\sigma} = \xi \sigma, \qquad \xi = e^{\tau \mathcal{N}(0,1)}. \tag{5}$$

where τ is called *learning parameter* and it is an exogenous parameter. In general we choose τ such that $0 \leq \tau \leq 1$.

2.3 Differential Evolution (DE/rand/1/bin)

DE, as other Evolutionary Algorithms, starts from an initial population of μ individuals randomly generated. Hence, the mutation and recombination operators are applied to generate descendants. There is not a reproduction phase because DE works with an asexual reproduction. Figure 4 shows the pseudo-code for DE.

```
DE Algorithm
Begin
    $g = 0$;
    initialize$\Big( \mathcal{B}_{\mu}^{(0)} = \Big\{ \Big( \mathbf{y}_m^{(0)},\ F(\mathbf{y}_m^{(0)}) \Big) \mid\ m = 1,..,\mu \Big\} \Big)$;
    Repeat
        For $m$=1 To $\mu$
            $\mathbf{y}_l$ =DE_mutation$\big(\mathbf{y}_m^{(g)}\big)$;
            $\tilde{\mathbf{y}}_l$ =DE_recombination$\big(\mathbf{y}_l\big)$;
            $\tilde{F}_l$ =$F(\tilde{\mathbf{y}}_l)$;
            $\Big(\mathbf{y}_m^{(g+1)}, F(\mathbf{y}_m^{(g+1)})\Big)$= selection$\Big[\Big(\tilde{\mathbf{y}}_l, \tilde{F}_l\Big), \Big(\mathbf{y}_m^{(g)}, F(\mathbf{y}_m^{(g)})\Big)\Big]$;
        End
        $g = g + 1$;
    Until $stop\_criterion$
End
```

Fig. 4. DE algorithm.

The fundamental characteristic of DE is a new way of generating descendants. For each descendant, we randomly choose an individual $\mathbf{y}_{r1}$ in the parents set, this vector is called *donor*. Summing to $\mathbf{y}_{r1}$ a new vector $\mathbf{z}$, obtained by the weighted difference of two individuals randomly selected by the population, we have that

$$\mathbf{y}_l = \mathbf{y}_{r1} + \mathbf{z} \qquad \text{with} \qquad \mathbf{z} = F(\mathbf{y}_{r2} - \mathbf{y}_{r3}) \tag{6}$$

where

$$r1, r2, r3 \in \{1, 2, \ldots, \mu\} \quad \text{such that} \quad r1 \neq r2 \neq r3 \neq m. \tag{7}$$

In the above equation F is an exogenous parameter usually fixed to 0.8, see [3] and [4].

In order to introduce innovation in the population genotype, the mutated vector $\mathbf{y}_l$ is recombined discretely and in not uniform way with the parameters vector associated to the current parent $\mathbf{y}_m$. The result is the vector $\tilde{\mathbf{y}}_l$ associated to l-th descendant whose components are calculated as shown in figure 5. In figure 5 *CR* is the crossover probability which determines the descendant vector components recombination while N is the genes number for every individual.

The DE/rand/1/bin is a standard variant of DE in which the donors are chosen randomly and discretely recombined with $\mathbf{y}_m$ using binomial distribution. The symbol 1 indicates the couples number of randomly chosen vectors contributing to make the mutation vector $\mathbf{z}$.

2.4 Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is a stochastic optimization technique developed by [3] and inspired by social behavior of bird flocking or fish schooling. *As stated before, PSO simulates the behavior of bird flocking. Suppose the following scenario: a group of birds are randomly searching food in an area. There is only one piece of food in the area being searched. All the birds do not know where the food is. But they know how far the food is in each iteration. The most effective strategy is to follow the bird which is nearest to the food.*

Considering an optimization problem, we can think at a possible solution as particles flying in an N-dimensional search space (schematized as an hypersphere) and everyone updates its position and speed considering both its experience and the neighbor one. The target is to land on the best solution.

PSO combines a *local search* with a *global search* of optimum.

PSO is initialized with a group of random particles (solutions) and it searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far (step by step fitness values for each individual are stored); this value is called $\mathbf{y}_{lbest}$. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population; this "best value" is a global best and it is called $\mathbf{y}_{gbest}$. After finding the two best values, the particle updates its "velocity" and "position" with the following equations:

$$\mathbf{v}^{(g+1)} = \mathbf{v}^{(g)} + c_1 \cdot \text{rand}[0,1] \cdot (\mathbf{y}_{lbest}^{(g)} - \mathbf{y}^{(g)}) + c_2 \cdot \text{rand}[0,1] \cdot (\mathbf{y}_{gbest}^{(g)} - \mathbf{y}^{(g)}) \quad (8)$$

$$\mathbf{y}^{(g+1)} = \mathbf{y}^{(g)} + \mathbf{v}^{(g+1)} \quad (9)$$

```
$j = \text{int}(rand[1, N]);$
For $i$=1 To $N$
    $\tilde{\mathbf{y}}_l(i) = \begin{cases} \mathbf{y}_l(i), & \text{if } (rand_i[0,1) < CR \;\vee\; \text{i=j}) \\ \mathbf{y}_m(i), & \text{otherwise.} \end{cases}$
End
```

Fig. 5. Recombination process

where c_1 and c_2 are two positive learning factors. The factor c_1 weights the individual performance respect to the past performances while c_2 weights the individual performance respect to the swarm.

Figure 6 shows the PSO pseudo-code.

```
PSO-Algorithm
Begin
    $g = 0$;
    initialize$\left( \mathcal{B}_{\mu}^{(0)} = \left\{ \left( \mathbf{y}_m^{(0)}, \mathbf{v}_m^{(0)}, F(\mathbf{y}_m^{(0)}) \right) \mid m = 1, .., \mu \right\} \right)$;
    Repeat
        For $m$=1 To $\mu$
            Calculate fitness: $F = F(\mathbf{y}_m)$;
            Find best fitness
            in particle story: $\mathbf{y}_{lbest}^{(g)}$;
        End
        Select in the population the particle
        with best fitness: $\mathbf{y}_{gbest}^{(g)}$;
        For $m$=1 To $\mu$
            Calculate speed: $\mathbf{v}_m^{(g+1)}$;
            Update position: $\mathbf{y}_m^{(g+1)}$;
        End
        $g = g + 1$;
    Until $stop\_criterion$
End
```

Fig. 6. PSO-Algorithm.

3 Stochastic Algorithm Computational Complexity Comparison: Test Set-Up

In this section, we explain how the experiment was driven.

We tried to compare the performances of the above described algorithms in searching optimal solution of a minimization problem on some *hard* test functions known in literature. The comparing criterion is based on the *computational complexity* which means the times the fitness function is calculated. In fact, in real optimization problems, the fitness function is often computationally onerous because it is a model of a real physical system; for example, it could be made up on integral-differential equations, so the repeated evaluation of these equations involves a large amount of time.

In our case, the computational complexity is equal to the product of individuals number and the minimum number of generations needful to find a stable solution of the problem.

3.1 Test Functions

As above stated, in order to compare algorithms performances we used a series of test functions. These ones are catalogued as *hard* in literature because they are multimodal and, in general, not differentiable. Follow the test functions:

1. Shubert function:

$$z = -\sum_{i=1}^{5} i \cdot cos[(i+1)x + i] \cdot \sum_{i=1}^{5} i \cdot cos[(i+1)y + i] \tag{10}$$

2. Shekel function:

$$z = -500 - \left(0.002 + \left(\sum_{i=1}^{25}(i + (x-a)^6 + (y-b)^6)\right)^{-1}\right)^{-1} \tag{11}$$

3. Pic function:

$$z = -13 \cdot e^{-\sqrt{(x+5.5)^2+(y+5.5)^2}} \tag{12}$$

4. Rosenbrock function:

$$z = (1-x)^2 + 100 \cdot (y - x^2)^2 \tag{13}$$

5. McCormic function:

$$z = sin(x+y) + (x-y)^2 - 1.5 \cdot x + 2.5 \cdot y + 1 \tag{14}$$

6. Hansen function:

$$z = -\sum_{i=1}^{5} i \cdot cos[(i-1)x + i] \cdot \sum_{i=1}^{5} i \cdot cos[(i+1)y + i] \tag{15}$$

Next figures show the shape for some test functions:

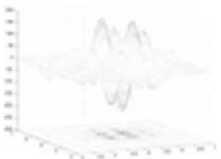


Fig. 7. Shubert function.

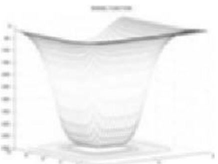


Fig. 8. Shekel function.

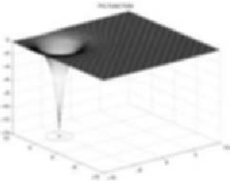


Fig. 9. Pic function.

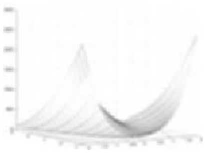


Fig. 10. Rosenbrock function.

3.2 Results

In the next figures we show the results of our tests. By these tests, we can state that the most performing algorithms is the PSO either for results accuracy or the convergence speed. Moreover, between the two kind of ES the $(1 + 1)$ version shows a better stability and speed than the $(1 + 5) - \sigma$SA. Also DE shows a good performance, in fact even if it converges for a great part of test functions later than PSO, it finds minimum with a better precision. In particular for Pic function DE seems to have better performances than PSO. For these reasons, we decided to create a new hybrid algorithm called *PSO-DE* described in the last part of this paper. In all the above figures, the fitness shape of the best individual vs the number of fitness function evaluations are shown.

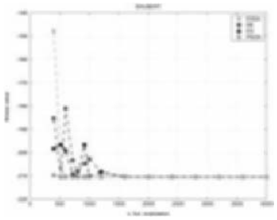


Fig. 11. "Shubert function" testing.

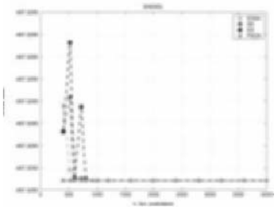


Fig. 12. "Shekel function" testing.

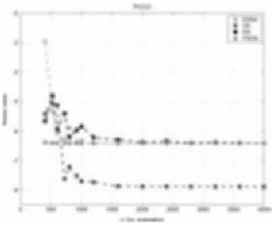


Fig. 13. "Pic function" testing.

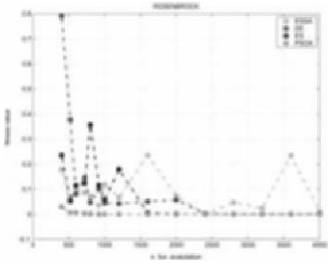


Fig. 14. "Rosenbrock function" testing.

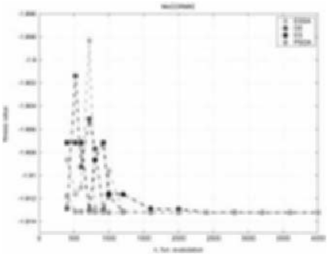


Fig. 15. "McCormic function" testing.

4 A New Hybrid Algorithm: PSO-DE

As explained in previous section, PSO and DE algorithms shows the best performance for the considered minimization problems.

We decided to merge the explorative attitude of PSO with differential mutation of DE as a source of innovation. Then the PSO-DE was born. From an algorithmic viewpoint, the novel algorithm works as follow:

- *Initial population is generated by PSO*
- *DE, starting by the above generated population, begins its natural evolution*

By this method we obtained a very fast convergence with an high accuracy. For example the simple PSO cannot reach the Pic function minimum while this algorithm goes deepest in the direction of absolute minimum. The simple PSO finds the Shubert function minimum in 15 iterations while the PSO-DE converges in about 4 iteration holding fixed the number of individuals in population. The next figures shows some results on test functions.

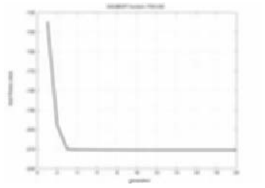


Fig. 16. "Shubert function" testing.

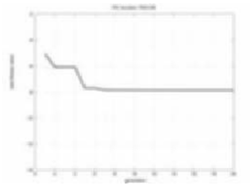


Fig. 17. "PIC function" testing.

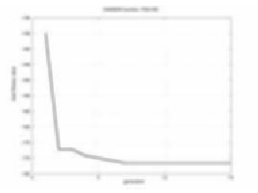


Fig. 18. "Hansen function" testing.

5 Conclusions

In this paper we compared four stochastic algorithms well known in literature: ES-(1+1), ES-(1+5)-σSA, PSO, DE. For ES-(1+1) we introduced an annealing logic to update the search step in order to avoid that our search falls in a local minimum. We compared these algorithms in searching the absolute minimum of some test functions catalogued as *hard* in literature because they are multimodal and not always differentiable. The comparing meter was the computational complexity. Our tests results suggest that PSO, for the analyzed minimum search problems, is the best algorithm. Also DE shows good performances. For these reasons, we developed a novel hybrid algorithm PSO-DE having better performances than other analyzed algorithms.

References

1. H. G. Beyer. *The Theory of Evolution Strategies.* Springer-Verlag, New York, 2001.
2. B. Naudts L. Kallel and A. Rogers. *Theoretical Aspects of Evolutionary Computing.* Springer-Verlag, New York, 2001.
3. R. Storn and K. Price. Differential evolution - a simple and efficient scheme for global optimization over continues spaces. *Technical Report TR-95-012 at Internal Computer Science Institute.*
4. M. Dorigo D. Corne and F. Glover. *New Ideas in Optimization.* McGraw-Hill International(UK), 1999.

Nonlinear Identification Method of a Yo-yo System Using Fuzzy Model and Fast Particle Swarm Optimisation

Bruno Meirelles Herrera[1], Leonardo Ribas[1] and Leandro dos Santos Coelho[2]

[1] *Cinq Technologies*
Rua da Glória, 72, 8º andar, Centro Cívico
CEP 80030-060, Curitiba, PR, Brazil
E-mail: bherrera@terra.com.br, lrribas@terra.com.br

[2] *Pontifícia Universidade Católica do Paraná, Grupo Produtrônica*
Programa de Pós-Graduação em Engenharia de Produção e Sistemas
Rua Imaculada Conceição, 1155, CEP 80215-901, Curitiba, PR, Brazil
E-mail: leandro.coelho@pucpr.br

Abstract. *Nonlinear complexities and unknown uncertainty of models for dynamical systems are difficult problems in identification tasks. Fuzzy systems, especially Takagi-Sugeno (TS) fuzzy systems, viewed as nonlinear systems are potential candidates for identification and control of general nonlinear systems. A method of nonlinear identification in open-loop based on TS fuzzy system is evaluated in this paper. The contribution of this paper is the proposition of an optimization approach to automatically build a TS fuzzy system based on a set of input-output data of a process. The proposed scheme is based in particle swarm optimization with operators based on Gaussian and Cauchy distributions for the antecedent part design, while least mean squares technique is utilized for consequent part of production rules of a TS fuzzy system. Experimental example using a nonlinear yo-yo motion control system is analyzed by proposed approach.*

Keywords: particle swarm optimisation, fuzzy systems.

1. Introduction

The knowledge of the behavior of real systems by using dynamic models is important in many fields of science and engineering. System identification can be described, as the art and the science of building mathematical models of dynamic system and signals based on observed inputs and outputs. The practical importance of process model identification has been recognized for many years [6].

Fuzzy systems viewed as nonlinear systems are potential candidates for modeling and control of general nonlinear systems. The Takagi-Sugeno fuzzy model [12],[11] exhibits both high nonlinearity and simple structure. As reported in the literature, it is capable of approximating a complex system using fewer fuzzy rules compared to conventional fuzzy models of Mamdani's type. The identification problem in Takagi-Sugeno (TS) modeling consists of two major parts, the structure identification and the parameter identification. Furthermore, the TS system comprises the premise part identification and the consequent part identification. Identification of the premise parte consists of determining the premise space partition and extracting the number of rules. The consequent part identification consists of determing the structure of the rules' output parts. Finaly, the parameter learnig task consists of determining the system parameters so that a performance measure based on the output errors is minimised. In this paper the structure identification of the premise and the consequent part are separately performed. The structure identification is realized based memetic algorithm for premise part optimization and the consequent part optimization is realized by least mean square method.

This paper presents the configuration of a TS fuzzy system based on optimization through a fast particle swarm optimization (premise part) and least mean squares (consequent) of a TS fuzzy system for identification of a yo-yo motion control prototype. The particle swarm optimization (PSO) algorithm belongs to the category of swarm intelligence methods. It was developed and first introduced by Kennedy and Eberhart [4]. During the last years, PSO gained increasing popularity due to its effectiveness in performing difficult optimization tasks. PSO is initialized with a population of random solutions. Its development was based on observations of the social behavior of animals such as bird flocking, fish schooling, and swarm theory. Each individual in PSO is assigned with a randomized velocity according to its own and its companions' flying experiences, and the individuals, called particles, are then flown through hyperspace. This paper proposes a technique of fast evolutionary programming [13],[1] to modification of velocity equation of PSO.

The next sections of this paper are organized in the following way. In the section 2 are described the TS fuzzy system fundamentals and it's otimization procedure. The procedure of nonlinear identification adopted is described in section 3. The prototype of yo-yo motion system and the results analysis applied to fuzzy system are shown in secton 4, respectively. Finally, conclusions are presented in section 5.

2. Takagi-Sugeno Fuzzy System

The TS fuzzy model is based on rules in which the consequent is not a linguistic variable, as in the Mamdani-type fuzzy model, but a function of the input variables. A relevant aspect of TS system is its power representation, especially for description of complex process. This fuzzy system allows complex system decomposition into simple sub-systems.

The identification of a TS fuzzy model involves two primary tasks: parameter tuning and structure optimization. The parameter tuning procedure deals with the estimation of a feasible set of parameters for a given structure. The structure optimization procedure aims to find the optimal structure of the local models, the relevant premise variables and a suitable partition of the premise space.

The TS models consist of linguistic IF-THEN rules that can be represented by the following general form:

$$\begin{aligned} &R^{(j)} : \mathit{IF}\ z_1\ \mathit{IS}\ A_1^j\ \mathit{AND} \ldots \mathit{AND}\ z_m\ \mathit{THEN} \\ &\qquad g_j = w_0^j + w_1^j u_1^j + \ldots + w_{qj}^j u_{qj}^j \end{aligned} \tag{1}$$

The IF preconditioned statements define the premise part while the THEN rule functions constitute the consequent part of the fuzzy system; $\underline{z} = [z_1, \ldots, z_m]^T$ is the input vector of the premise p, A_i^j and are labels of fuzzy sets. The parameters $\underline{u} = [u_1^j, \ldots, u_{qj}^j]^T$ represents the input vector to the consequent part of $R^{(j)}$ that comprising q_j terms; $g_j = g_j\left(\underline{u}^j\right)$ denotes the *j*-th rule output which is a linear polynomial of the consequent input terms u_i^j, and $\underline{w}^j = [w_0^j, w_1^j \ldots, w_{qj}^j]^T$ are the polynomial coefficients that form the consequent parameter set. Each linguistic label A_i^j is associated with a membership function, $\mu_{A_i^j}(z_i)$, which described by

$$\mu_{A_i^j}(z_i) = \exp\left[-\frac{1}{2}\frac{\left(z_i - m_{ij}\right)^2}{\sigma_{ij}^2}\right] \tag{2}$$

where m_{ij} and σ_{ij} are the mean value and the standard deviations of the Gaussian type membership function, respectively. The union of all these parameters formulates the premise parameter set. The firing strength of rule $R^{(j)}$ represents its excitation level and it is given by:

$$\mu j(\underline{z}) = \mu_{A_1^j}(z_1) \cdot \mu_{A_2^j}(z_2) \cdots \mu_{A_m^j}(z_m) \tag{3}$$

The fuzzy sets pertaining to a rule form a fuzzy region (cluster) within the premise space, $A_1^j \, x \, A_2^j x \cdots x \, A_m^j$, with a membership distribution described by equation (3). Given the input vectors $\underline{z}$ and $\underline{u}^j$, $j = 1, \ldots,, M$, the final output of the fuzzy system is inferred by taking the weighted average of the local outputs $g_j\left(\underline{u}^j\right)$ that is given by

$$y = \sum_{j=1}^{M} v_j(\underline{z}) \cdot g_j\left(\underline{u}^j\right) \tag{4}$$

where M denotes the number of rules and $v_j(\underline{z})$ is the normalized firing strength of $R^{(j)}$, which is defined as

$$v_j(\underline{z}) = \frac{\mu_j(\underline{z})}{\sum_{j=1}^{M} \mu_j(\underline{z})} \tag{5}$$

The structure identification of TS system is realized based fast PSO for premise part optimization and the consequent part optimization is realized by least mean squares method [6]. The PSO procedure for fuzzy model optimization is presented in the next section.

3. PSO for Optimization of TS Fuzzy System

PSO is a stochastic optimisation technique and has been shown to be effective in optimising difficult multidimensional discontinuous problems in a variety of fields. The PSO is motivated from the simulation of social behaviour instead of the evolution of nature as in the other swarm intelligence approaches (ant systems and bacterial colony) and evolutionary algorithms.

In PSO algorithm the population dynamics simulates a bird flock's behaviour where social sharing of information takes place and individuals can profit form the discoveries and previous experience of all other companions during the search for food. Thus, each companion, called *particle,* in the population, which now called *swarm*, is assumed to "fly" over the search space in order to find promising regions of the landscape. In this context, each particle is treated as a point in an *n*-dimensional space which adjusts its own "flying" according to its flying experience as well as the flying experience of other particles (companions) [7].

PSO is a population-based evolutionary algorithm. Similar to the other population-based evolutionary algorithms, PSO is initialised with a population of random solutions. Unlike the most of the evolutionary algorithms, each potential solution (individual) in PSO is also associated with a randomised velocity, and the potential solutions (*particles*), are then "flown" through the problem space.

Each particle keeps track of its coordinates in the problem space, which are associated with the best solution (fitness) it has achieved so far. This value is called *pbest*. Another "*best*" value that is tracked by the *global* version of the particle swarm optimiser is the overall best value, and its location, obtained so far by any particle in the population. This location is called *gbest.*

The PSO concept consists of, at each time step, changing the velocity (accelerating) of each particle flying toward its *pbest* and *gbest* locations (global version of PSO). Acceleration is weighted by random terms, with separate random numbers being generated for acceleration toward *pbest* and *gbest* locations, respectively. The procedure for implementing the global version of PSO is given by the following steps [5]:

(i) Initialise a population (array) of particles with random positions and velocities in the *n* dimensional problem space.

(ii) For each particle, evaluate its fitness value.

(iii) Compare each particle's fitness evaluation with the particle's *pbest*. If current value is better than *pbest*, then set *pbest* value equal to the current value and the *pbest* location equal to the current location in *n*-dimensional space.

(iv) Compare fitness evaluation with the population's overall previous best. If current value is better than *gbest*, then reset *gbest* to the current particle's array index and value.

(v) Change the velocity and position of the particle according to equations (6) and (7), respectively [10]:

$$\boldsymbol{v_i} = w \cdot \boldsymbol{v_i} + c_1 \cdot ud(\,) \cdot (\boldsymbol{p_i} - \boldsymbol{x_i}) + c_2 \cdot Ud(\,) \cdot (\boldsymbol{p_g} - \boldsymbol{v_i}) \tag{6}$$

$$\boldsymbol{x}_i = (\boldsymbol{x}_i + \Delta t \cdot \boldsymbol{v}_i) \ . \tag{7}$$

where Δt is chosen to be one.

(iv) Loop to step (ii) until a criterion is met, usually a sufficiently good fitness or a maximum number of iterations (generations).

where $\boldsymbol{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^{\mathrm{T}}$ stands for the position of the *i*-th particle, $\boldsymbol{v}_i = [v_{i1}, v_{i2}, \ldots, v_{in}]^{\mathrm{T}}$ stands for the velocity of the *i*-th particle and $\boldsymbol{p}_i = [p_{i1}, p_{i2}, \ldots, p_{in}]^{\mathrm{T}}$ represents the best previous position (the position giving the best fitness value) of the *i*-th particle. The index *g* represents the index of the best particle among all the particles in the group. Variable *w* is the inertia weight, c_1 and c_2 are positive constants; *ud*() and *Ud*() are two random functions in the range [0, 1]. Particles' velocities on each dimension are clamped to a maximum velocity *Vmax*. If the sum of accelerations would cause the velocity on that dimension to exceed *Vmax*, which is a parameter specified by the user, then the velocity on that dimension is limited to *Vmax*.

Vmax is an important parameter. It determines the resolution with which the regions around the current solutions are searched. If *Vmax* is too high, the PSO facilitates global search, and particles might fly past good solutions. If *Vmax* is too small, on the other hand, the PSO facilitates local search, and particles may not explore sufficiently beyond locally good regions.

The first part in equation (6) is the momentum part of the particle. The inertia weight *w* represents the degree of the momentum of the particles. The second part is the "cognition" part, which represents the independent thinking of the particle itself. The third part is the "social" part, which represents the collaboration among the particles. The constants c_1 and c_2 represent the weighting of the "cognition" and "social" parts that pull each particle toward *pbest* and *gbest* positions.

3.1. Fast Particle Swarm Optimisation

In this paper, new approaches to PSO, named fast PSO are proposed which are based on the studies of mutation operators in fast evolutionary programming) [13], [1]. The aim is to modify the equation (6) of the conventional PSO (case 1) to use it with Gauss or Cauchy distribution. The use of Cauchy distribution in evolutionary algorithms could be useful to escape of local minima, while the Gauss distribution (normal distribution could provide a faster convergence in local searches [8]. The modification of the equation (6) (conventional PSO) proceeds as follows:

Type 2: It is used a function with Cauchy distribution, *cd()*, to generate random numbers in the interval [0, 1] for the "cognitive part":

$$\boldsymbol{v_i} = w \cdot \boldsymbol{v_i} + c_1 \cdot cd(\,) \cdot (\boldsymbol{p_i} - \boldsymbol{x_i}) + c_2 \cdot Ud(\,) \cdot (\boldsymbol{p_g} - \boldsymbol{v_i}) \tag{8}$$

Type 3: It is used a function with Cauchy distribution *Cd()*, to generate random numbers in the interval [0, 1] for the "social part":

$$\boldsymbol{v_i} = w \cdot \boldsymbol{v_i} + c_1 \cdot ud(\,) \cdot (\boldsymbol{p_i} - \boldsymbol{x_i}) + c_2 \cdot Cd(\,) \cdot (\boldsymbol{p_g} - \boldsymbol{v_i}) \tag{9}$$

Type 4: It is used a function with Cauchy distribution, *cd()*, and *Cd(),* to generate random numbers in the interval [0, 1] for the "cognitive part" and "social part":

$$\boldsymbol{v_i} = w \cdot \boldsymbol{v_i} + c_1 \cdot cd(\,) \cdot (\boldsymbol{p_i} - \boldsymbol{x_i}) + c_2 \cdot Cd(\,) \cdot (\boldsymbol{p_g} - \boldsymbol{v_i}) \tag{10}$$

Type 5: It is used a function with Gauss distribution, *gd()*, to generate random numbers in the interval [0, 1] for the "cognitive part":

$$\boldsymbol{v_i} = w \cdot \boldsymbol{v_i} + c_1 \cdot gd(\,) \cdot (\boldsymbol{p_i} - \boldsymbol{x_i}) + c_2 \cdot Ud(\,) \cdot (\boldsymbol{p_g} - \boldsymbol{v_i}) \tag{11}$$

Type 6: It is used a function with Gauss distribution *Gd()*, to generate random numbers in the interval [0, 1] for the "social part":

$$\boldsymbol{v_i} = w \cdot \boldsymbol{v_i} + c_1 \cdot ud(\,) \cdot (\boldsymbol{p_i} - \boldsymbol{x_i}) + c_2 \cdot Gd(\,) \cdot (\boldsymbol{p_g} - \boldsymbol{v_i}) \tag{12}$$

Type 7: It is used a function with Gauss distribution, *gd()* and *Gd()*, to generate random numbers in the interval [0, 1] for the "cognitive part" and "social part":

$$\boldsymbol{v_i} = w \cdot \boldsymbol{v_i} + c_1 \cdot gd(\,) \cdot (\boldsymbol{p_i} - \boldsymbol{x_i}) + c_2 \cdot Gd(\,) \cdot (\boldsymbol{p_g} - \boldsymbol{v_i}) \tag{13}$$

Type 8: It is used a function to generate random numbers in the interval [0, 1] with Gauss distribution, *gd()*, for the "cognitive part" and with Cauchy distribution for the "social part":

$$\boldsymbol{v_i} = w \cdot \boldsymbol{v_i} + c_1 \cdot gd(\,) \cdot (\boldsymbol{p_i} - \boldsymbol{x_i}) + c_2 \cdot Cd(\,) \cdot (\boldsymbol{p_g} - \boldsymbol{v_i}) \tag{14}$$

Type 9: It is used a function to generate random numbers in the interval [0, 1] with Cauchy distribution, *cd()*, for the "cognitive part" and with Gauss distribution for the "social part":

$$\boldsymbol{v_i} = w \cdot \boldsymbol{v_i} + c_1 \cdot cd(\,) \cdot (\boldsymbol{p_i} - \boldsymbol{x_i}) + c_2 \cdot Gd(\,) \cdot (\boldsymbol{p_g} - \boldsymbol{v_i}) \qquad (15)$$

4. Yo-yo Motion Process

The development of automatic control systems that control efficiently a yo-yo represents a significant challenge for the development of electromechanical projects [2]. One of the main difficulties is the lack of sensors to obtain the motions measures of the toys. Another difficulty is the lack of mathematical models of this measurement device type, what it justifies the use of the TS fuzzy system for identification of the dynamic behavior of a yo-yo motion real system.

The prototype of control system uses a yo-yo and a direct current (DC) motor for its motion presents nonlinearity and complex behavior. The motivation for the use of this process is the possibility to prove the efficiency and flexibility of soft computing and adaptive control systems. A blocks diagram of the described system and a photograph of the system are presented in the figures 1 and 2, respectively [3]. The components of this prototype are divided in software and hardware modules, where:

① *Control module (software)*: consists of implementation of control techniques, such as PID (proportional, integral and derivative) control, fuzzy logic control, and PI adaptive controllers integrated into computer with communication to the yo-yo system using an I/O interface;

② *Sensor module (hardware / firmware)*: the sensors employed include the digital electronic circuits (power amplification), A/D and D/A converters, and the micro controller running firmware;

③ *Actuator module (hardware / firmware)*: consists of DC motors integrated with the *Sensor module*, electronic circuits and micro controller running firmware;

④ *Sensor sub-module*: made up with 16 infrared *LEDs* able to inform the position of the yo-yo.

The prototype modules are composed of hardware and firmware and had been connected to the same printed circuit board called control board. The control board contains two hardware modules and its communicate with a personal computer (*Main module*) through the RS-232 I/O interface. All of components used for the yo-yo system are off the shelf items to keep the cost minimal. The prototype can serve as tangible evidence of the usefulness of nonlinear identification approaches and self-tuning, predictive and fuzzy control techniques in difficult situations.

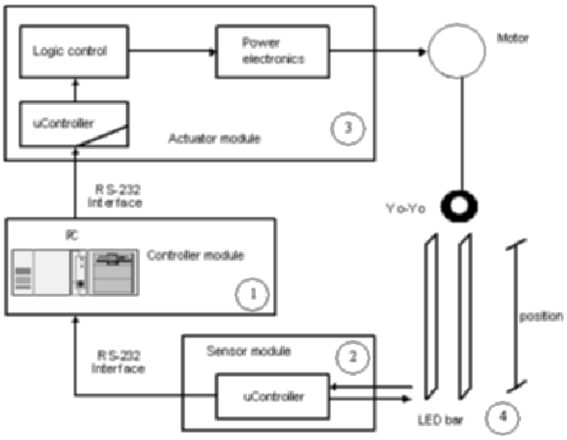


Fig. 1. Block diagram of yo-yo motion system.

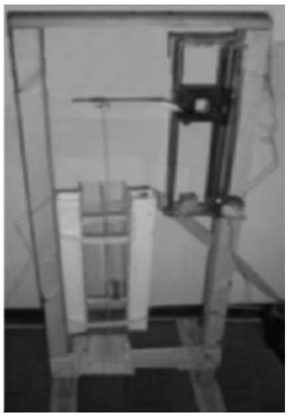


Fig. 2. Photo of prototype of yo-yo motion system.

5. Analysis Results of Identification

The goal of the identification of systems is to allow the adjustment of a mathematical model for a dynamic system, based on measurements collected by the adjustment of parameters and/or of the model, until the system output close "so well", as possible, to the samples of the measured outputs. The procedure for experimental identification of processes is constituted by four basic stages, according to Figure 3.

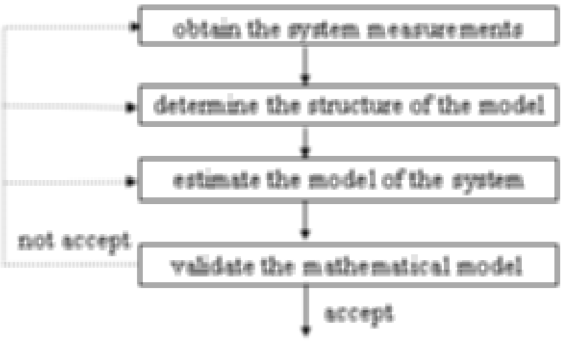

obtain the system measurements
determine the structure of the model
estimate the model of the system
not accept
validate the mathematical model
accept

Fig. 3. Procedure for identification of yo-yo motion system.

A computer with data acquisition board for generation of the control signal (identification in closed-loop using a proportional controller) and position value of the yo-yo were used for obtaining of system measurements. In identification procedure based on TS fuzzy model had been collected 290 samples of input (tension applied to the DC motor) and output (position of yo-yo) with time sampling of 40ms (see figure 4). The tension value corresponds to the maximum value configuration of driver in PWM (pulse width modulation) control of motor.

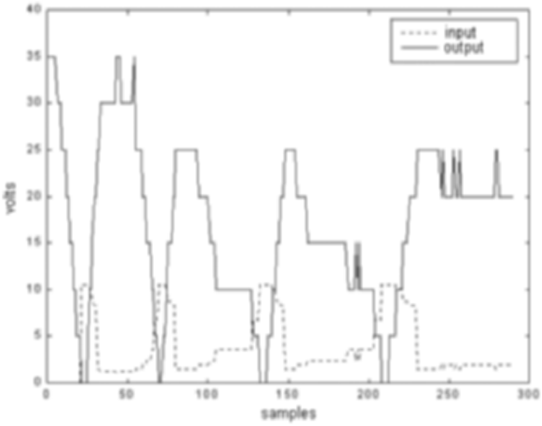


Fig. 4. Prototype's inputs and outputs data.

Experiments for the estimation phase of the mathematical model of yo-yo motion system are carried out using the samples 1 to 150. For the validation phase, the fuzzy model uses input and output signals of the samples 151 to 290.

The system identification by TS fuzzy model is appropriate if a performance index is in permissible values to the necessities of user. The chosen performance index (function of fitness) was the multiple correlation index conducted by

$$R^2 = 1 - \frac{\sum_{t=1}^{Na} (y(t) - \hat{y}(t))^2}{\sum_{t=1}^{Na} (y(t) - \bar{y})^2}, \tag{16}$$

where *Na* is the number of samples evaluated, $y(t)$ is real output of the system, $\hat{y}(t)$ is the estimated output by the TS fuzzy system, $\bar{y}$ is the system real output. When the value R^2=1.0 it indicates an accurate approach of the model for the measured data of the system. A R^2 between 0.9 and 1.0 it is considered enough for applications in projects of identification and model-based control [9]. As presented in table 1, the results of the TS fuzzy system using serie-parallel structure presents precision and provided an appropriate experimental mathematical model for yo-yo motion system. TS fuzzy model presents a black-box model of yo-yo system, with treatment adequate of the non-linearities of the dynamic system due to inherent features of fuzzy systems to deal with complex processes. Based on previous experience with particle swarm optimisation (trial and error, mostly) led us to set the acceleration constants c_1 and c_2 equal to 2.0; *Vmax* set to 20% of the dynamic range of the variable on each dimension (chosen dimension in the range [-10, 10]) and the number of particles to 30 and 100 generations (stop criterion) for the simulations with the TS fuzzy model optimisation. The three chosen vectors of input for the TS fuzzy system were [$u(t)$; $u(t$-1); $y(t$-1)].

Table 1. Summary of the results (better of 50 experiments) for different number of Gaussian membership functions (*mf*) of TS fuzzy system.

best PSO type	*mf*	R^2 (estimation)	R^2 (validation)	best PSO type	*mf*	R^2 (estimation)	R^2 (validation)
9	2	0.4976	0.4723	9	10	0.9594	0.9351
5	3	0.6221	0.7098	5	**11**	**0.9841**	**0.9872**
9	4	0.9333	0.8842	9	12	0.9673	0.9412
5	5	0.9484	0.9061	5	13	0.9671	0.9536
9	6	0.9544	0.9312	2	14	0.9727	0.9775
9	7	0.9562	0.9398	9	15	0.9848	0.9889
8	8	0.9519	0.9392	9	16	0.9844	0.9861
9	9	0.9534	0.9351	9	17	0.9766	0.9843

It is noticed from table 1, that for *mf* >5 all the results of TS fuzzy model gotten had been with $R^2 > 0.90$, that is, necessary and therefore appropriate for controllers project applications. In figure 5 the best resulted (*mf*=11) gotten is presented. The best PSO for the experiments was the PSO of type 9 with 10 best results in table 1.

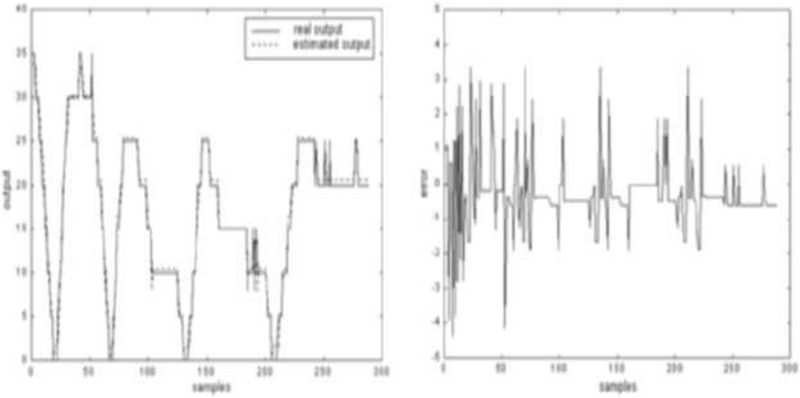


Fig. 5. Best result (with *mf*=11 from table 1) in yo-yo motion system identification.

6. Conclusion and Future Works

In this paper, an approach of optimisation of a TS fuzzy systems was proposed. The optimisation approach of the premise part of the rules was based on fast PSO, while the optimization of the consequent part of the rules was conducted by least mean squares method.

The experimental results had shown that the TS fuzzy system with fast PSO presented potentialities due precision in the prediction of nonlinear dynamics of a yo-yo motion system. However, the analysis of one better analysis about the precision, computational complexity and orders of the input vectors of TS fuzzy model must be evaluated with more details in future works. As perspectives of future works linked to the yo-yo motion system design there is a possibility to assess a comparative of other tools in identification and control, being adopted intelligent, self-tuning and adaptive methodologies.

References

1. Chellapilla, K. (1998) Combining mutation operators in evolutionary programming, *IEEE Trans. Evolutionary Computation* 2(3), pp. 91-96
2. Hashimoto, K.; Toshiro, N. (1996) Modeling and control of robotic yoyo with visual feedback, *IEEE International Conference on Robotics and Automation*, Minneapolis, Minnesota, pp. 2650-2655

3. Herrera, B.M.; Ribas, L.V. (2003) Artificial intelligence approaches applied to the control of yo-yo motion system, Project of Conclusion Course in Computer Engineering, Pontifical Catholic University of Parana, Curitiba, PR, Brazil (in Portuguese)
4. Kennedy, J.; Eberhardt, R.C. (1995) Particle swarm optimization, *Proceedings of IEEE Conference on Neural Networks IV*, Piscataway, NJ
5. Krohling, R.A.; Coelho, L.S.; Shi, Y. (2002) Cooperative particle swarm optimisation for robust control system design, *7th Word Conference on Soft Computing in Industrial Applications*, Granada, Spain
6. Ljung, L. (1987) *System identification: theory for the user*, Prentice Hall, New York
7. Magoulas, G.D.; Eldabi, T.; Paul, R.J. (2002) Global search strategies for simulation optimisation, *Proc.Winter Simulation Conference*, Yücesan , E; Chen, C. -H.; Snowdon, J.L.; Charnes, J. M. (eds.), pp. 1978-1985
8. Rudolph, G. (1997) Local convergence rates of simple evolutionary algorithms with Cauchy mutations, *IEEE Transactions on Evolutionary Computation* 1, pp. 249-258
9. Schaible, B.; Xie, H.; Lee, Y.C. (1997) Fuzzy logic models for ranking process effects, *IEEE Transactions on Fuzzy Systems* 5(4), pp. 545-556
10. Shi, Y.; Eberhart, R.C. (1999) Empirical study of particle swarm optimisation, *Proceedings of the Congress on Evolutionary Computation*, Piscataway, NJ: IEEE Service Center, pp. 1945-1950
11. Sugeno, M.; Kang, G.T. (1988) Structure identification of fuzzy model, *Fuzzy Sets and Systems* 28, pp. 15-33
12. Takagi, T.; Sugeno, M. (1985) Fuzzy identification of systems and its applications to modeling and control, *IEEE Trans. Systems, Man and Cybernetics* 15(1), pp. 116-132
13. Yao, X.; Liu, Y. (1996) Fast evolutionary programming, *Proceedings of 5th Conference on Evolutionary Programming*, San Diego, CA, pp. 451-460

Part VIII

Manufacturing and Production

Hybrid Type-1-2 Fuzzy Systems for Surface Roughness Control

Fuhua Jiang*, Zhiyi Li and Yan-Qing Zhang, *Member, IEEE*

Department of Computer Science, Georgia State University
Atlanta, GA, 30303, USA
*: Contact author: fjiang2@gsu.edu

Abstract. A hybrid of type-1 and type-2 fuzzy model is proposed, which is applied in controlling the surface roughness of mechanical workpiece in metal cutting manufacturing. There are dozens of factors that affect the quality of surface roughness. The factors can be divided into two groups that are controlled and uncontrolled factors, e.g. feed rate can be setup. Therefore it is controlled factor while tool wear is an example of uncontrolled factor. There are two kinds of factors respectively correspond to type-1 and type-2 solutions because type-1 is suitable for controlled factors and type-2 fuzzy logic can handle uncontrolled or uncertain inputs. The proposed study will use genetic algorithm to identify the significant factors during the cutting process and a mathematical model that can predict the surface roughness under process variations. A fuzzy set based model for metal cutting operations can be used to reliably predict surface roughness under variations so that a continuous control of surface roughness can be affirmed. Two main factors (feed rate and tool wear) which affect the quality of surface roughness are investigated and simulated. The result of simulation shows that hybrid fuzzy logic system has improved precision of output.

Keywords: hybrid fuzzy control, type-2, genetic algorithm, surface roughness

1 Introduction

The globalization of market and increasing competition has made productivity of manufacturing systems and the quality of manufactured goods more critical for producers than before. Traditionally, the quality of a machined product is measured and inspected on the specifications of the products, after the machining process is complete. Controlling the quality of products in advance is a key to increasing the productivity and decreasing costs. Surface roughness can be specified in many different parameters. These parameters include Roughness average (R_a), root-mean-square (rms) roughness (R_q), and etc. Roughness average (R_a) is the mostly used international parameter of roughness (C. C. C. M. L. Mike S. Lou, 1999).

$$R_a = \frac{1}{L}\int_0^L |Y(x)|dx \tag{1}$$

Where R_a = Roughness average, ,L = the sampling length, Y = the ordinate of the profile curve. Root-mean-square (rms) roughness (R_q) is the root-mean-square parameter corresponding to R_a

$$R_q = \sqrt{[\frac{1}{L}\int_0^L (Y(x))^2 dx]} \tag{2}$$

Surface roughness, which is mainly used to measure the surface quality as Roughness Average (Ra), is playing an important role in evaluating the quality of mechanical products. Surface roughness is harder to manage than other physical dimensions because too many factors affect surface roughness. Factors influencing surface integrity are temperatures generated, design processing, residual stresses, metallurgical transformations, and surface plastic deformation, tearing, and cracking (S.Kalpakjian, 1997). Some factors can be controlled while other factors cannot be controlled. Controllable parameters include feed rate, cutting speed, tool geometry, and tool setup. Other parameters, such as machine vibration, tool wear, and tool variability can not be easily controlled (S. A. S. Coker, Y.C., et al. 1996). Some factors e.g., tool nose radius, temperature of machines in the tuning operation are varying from time to time.

To date, matching process models represent a mapping of input and output variables for specific machining condition. The selection of machine parameters depends on the experience of the process engineers when variations are included into the process. Empirical modeling has been popularly used because the formulation of the model is easy and the accuracy of the model is good. However, variations in work materials can cause significant fluctuations in the surface roughness produced. At the same time a machining process output may drift as time changes.

There is a multiple regression modeling analysis to determine the dependency of surface roughness R_a to selected machining parameters (Luke Huang, et al., 2001). The model was expressed as:

$$R_a = \beta_0 + \beta_s S + \beta_F F + \beta_D D + \beta_V V + \beta_{SF} SF + \beta_{SD} SD + \beta_{SV} SV + \beta_{DF} DF + \beta_{DV} DV + \beta_{FV} FV + \beta_{SDF} SDF + \beta_{SDV} SDV + \beta_{SFV} SFV + \beta_{DFV} DFV + \beta_{SDFV} SDFV \quad (3)$$

where R_a = surface roughness average, F = feed rate, S = spindle speed, D = depth of cut V = vibration amplitude average, B = linear constants. This model was developed for an on-line, real-time surface roughness prediction system. Using training data, the multiple regression models are integrated in the prediction system of surface roughness.

Reza Langari et al. have investigated the diagnosis of critical tool wear in machining of metals via a neuro-fuzzy algorithm. An adaptive fuzzy logic algorithm is established. The adaptive algorithm proves to improve the performance of predicting the condition of the tool in the milling process. (Thomas Hessburg et al.,1993) have investigated that the tuning of a fuzzy logic controller by genetic algorithm in an automated highway system. The genetic algorithm simultaneously determines the shape of membership functions, number of rules, and consequent parameters of fuzzy logic system.

There are two major input parameters that influence the output in surface roughness quality control: feed rate (F) and tool wear (TW). The output is surface roughness (Ra). These parameters generally are in range and not a single value because of the variation and limitation of machine. According to a different size of tool or different workpiece there are different feed rate. The feed rate and tool wear are the input parameters in our study of fuzzy logic systems. Feed rate falls down into controlled parameter category while tool wear in uncontrolled. The software in our project provides the ability to implement many-to-many model so as to extend the system easily.

A fuzzy logic system (FLS) is a control system that is able to handle numerical data and linguistic knowledge together. In general it is a nonlinear mapping of input data to a scalar output data. It includes fuzzifier, rules, inference engine, and defuzzifier (J. M. Mendel, 2001). The rules may be derived from person's knowledge or historical real data. The format of rules is a set of IF-THEN statements. How to adjust or refine the FLS is a challenge. One way depends on the experts to refine the output of FLS. However experts' knowledge mainly discovers the mechanism in the system, which has already been represented in the rules base. If experts can refine the output, we can also embed refinements into some rules. This required us to use FLS independent method to refine system to achieve high performance. In this system, the genetic algorithm that is a natural way to trace the time series information provides the refinement mechanism. The shapes and positions of fuzzy variables are refinement goals that genetic algorithm wants to modify because genetic algorithm converges all variables very quickly although its accuracy is not very high. The rest of the paper is organized as follows. Section 2 describes how type-1-2 matches two groups of factors based on understanding of processes of manufacturing. Section 3 presents how to carry out type-reducer and defuzzifier. In the section 4, an example is given to illustrate the performance of hybrid FLS. Finally, Section 5 gives conclusions.

2 Method

2.1 Fuzzy Knowledge Discovery

When focusing on the main factors, tool wear and feed rate, which affect the quality of surface roughness, we found that feed rate is relative easy to control and adjust through the tool machine while tool wear is almost impossible to measure during the manufacturing process. Fortunately, the value of tool wear can be measured by testing in the laboratory condition, however, the measurement involves some noisy and uncertain because it is not real manufacturing condition. On the other hand, the consequent surface roughness measuring also has a range of values and not just a single value or crisp interval.

To address this problem, a fuzzy logic control system is a natural way to handle those problems with uncertain conditions. However traditional fuzzy logic (type-1) system deals with fuzzy variables which have a range of values but the value is precise once measured; and linguistic meaning of words is not certain such that the same words may mean different things to different people. In the feed back based automatic control system model, the key elements of a control variable are output value measuring, input adjustment mechanism and adjustment algorithm. In terms of whether measuring is precise or not and whether adjustment is easy or not, we divided all factors that affect the results into two groups named group-1 and group-2. Group-1 is relatively easy to measure and control and group-

2 is difficult to measure and control. Therefore the feed rate falls into group-1 while tool wear falls into group-2. Clearly the type-1 system is suitable for group-1 variables. Type-1 fuzzy logic system cannot handle such uncertainty because their membership function is crisp. Now type-2 fuzzy logic system (N.N.M.Karnik, J.M. Qilian Liang, 1999) is introduced to deal with group-2 factors. The main reason is that there are four sources of uncertainties associated with fuzzy logic system type-1 system cannot be figured out well (J. M. Qilian Liang Mendel, 2000). 1. The meaning of words used in rules both condition and consequence are uncertain, which is from different people's different explanation about the same words. 2. Consequences may be hard to measure and describe precisely or there exists several methods of measurement. 3. The noise in the measurements is inevitable. 4. The data of tuning control system parameters may be also noisy. Type-2 fuzzy system can obviously solve group-2 variables such as tool wear variable in this system. Although the tool wear has somewhat certain linguistic meaning, the difficulty of measurement of tool wear is in that one piece of tool usually machines one more than work pieces. And there are no optimal methods to describe or measure the degree of tool wears. The variable feed rate is determined by the electromotor. The principle of those type-2 fuzzy sets can model such uncertainty because their membership functions are fuzzy themselves. This means its output can be a random value of a range. In summary, type-1 and type-2 fuzzy logic system can be named a hybrid system that is used in surface roughness control systems.

Genetic Algorithm(GA), an effectively approach in searching complex, nonlinear and multidimensional search space, for fuzzy rule shall be used for finding and modifying appropriate fuzzy rules and membership functions because the GA does not need to know anything about the problem domain (John Koza,1995). The parameters in surface roughness control system are in a range, not a crispy output. The input parameters used in this system are feed rate (F) and tool wear (TW). The block diagram of a type-2 fuzzy logic system is depicted in Fig. 1. In this figure the difference between type-2 and type-1 fuzzy logic system is type-2's degree of membership function is a fuzzy set. A type-2 fuzzy logic system can add uncertainties about measurements, fuzzy rules, consequent choices, and unreliable training data together into its outputs.

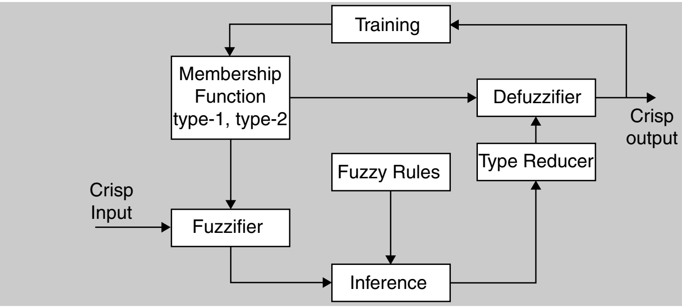


Fig. 1. Hybrid Fuzzy Logic System

Type-2 fuzzy logic system can be easily visualized to begin with a type-1 fuzzy logic system membership function. For example if a type-1 fuzzy logic system's membership function is a triangle, just blur it by shifting it left and right. Then input and output parameters in membership function are not crispy value, they are values in a range.

Many surface roughness control models for turning operations are based on experimental data. Empirical models are limited to a narrow domain and are very sensitive to the process variations so that even well defined empirical models may become inaccurate under process variations. A machining process output may drift and the surface roughness of turned parts becomes different than the values predicted by empirical equations. The empirical models have no means of incorporating changes in the process or responding to the process variations. Therefore, some classic type-1 fuzzy logic systems were introduced on this area. Now we would extend the type-1 system into a hybrid system since the main attributes of affecting surface roughness such as tool wear, feed rate are hard to measure and they are uncertain. We designed hybrid fuzzy logic system software, which accepts the type-1 or type-2 fuzzy input and output sets, as our test bed to test a 2-input and 1-ouput FLS. The two inputs are feed rate and tool wear in the manufacturing system, and the output is a singleton of surface roughness.

2.2 Type Reducer and Iterative Evolutionary Learning

FLS is consisted of at least 4 components such as fuzzifier, inference, defuzzifier and rules. In the type-2 FLS, the new component type-reducer is extended to deal with interval output sets, as shown on Fig. 2. Now we want to extend the legacy type-1 FLS into type-2. The membership functions (MF) used here are in triangle shape.

Every linguistic word has 3 values in type-1, which represent the triangle of left point x_1, middle point *m* and right point x_2 on the x-axis. The MF in the middle point is 1.0, while MF is 0.0 on the point of x_1 and x_2. Note that the output Ra is singleton which is the special case of type-1 where x_1=m=x_2

We take the triangle shape of fuzzy set as an example to illustrate the algorithm. Type-2 is comprised of inner and outer two triangles. The outer is determined by x1, m, and x2 while the inner is by x3, m, x4, as shown on Fig. 2. The type-1 is shown on Fig. 3.

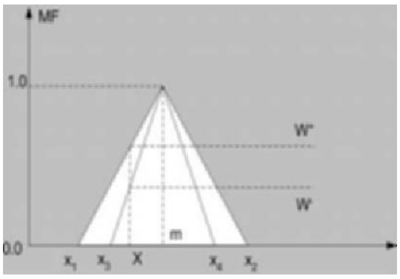


Fig. 2. Type-2

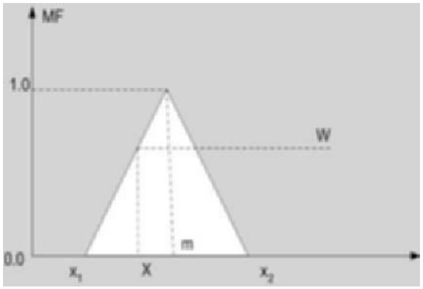


Fig. 3. Type-1

Obviously, the type-1 and type-2 are convertible. When $x_1=x_3$ and $x_2=x_4$, the output of type-2 w^+ and w^- will be overlapped as $w^+ = w^- = w$. In the other hand, the type-1 can be regarded as a special case of type-2. As the result, the T-Norm and S-Norm of type-1 and type-2 can be computed according to Mendel's on his paper (N.N. K. a. J. M. Mendel, 2001).

$$T\text{-}Norm(w^+, w^-) = (\, min(w^+_1, w^+_2,..., w^+_n), min(w^-_1, w^-_2,..., w^-_n)\,)$$

$$S\text{-}Norm(w^+, w^-) = (\, max(w^+_1, w^+_2,..., w^+_n), max(w^-_1, w^-_2,..., w^-_n)\,)$$

where n is the number of fuzzy sets of a fuzzy variable. In the type-reducer component, the type reduced set is:

$$f\,(w^+, w^-) = \alpha(1- s/S) + (1- \alpha)(w^+ + w^-)/2,$$

where: s is the segment area between w^+ and w^-, S is the total area of triangle, α is coefficient between $0<=\alpha<=1.0$

To simplify the computation of output of inference, we take the average of w^+ and w^- as the output. However, the average value will ignore some important information. For example, the average of the following of two groups of data is the same.

$w^+ = 0.8, w^- = 0.2$ and $w^+ = 0.6, w^- = 0.4$

Note that the larger S, the much fuzzier of output of inference. To overcome the method of averaging, we introduced a new factor segment area s/S to compensate the output. Again, to simplify the computation of area s/S, we would use $w^+ - w^-$ value as S.

$$f\,(w^+, w^-) = \alpha(1- w^+ + w^-) + (1- \alpha)(w^+ + w^-)/2,$$

When α=0, $w^+ = w^-$, f (w^+, w^-) = f(w) = w. therefore this formula is also suitable for type-1 fuzzy sets.

Manufacturing procedure is the process of evolvement, so the fuzzy logic model needs to be adjusted frequently. Genetic algorithm is rapid way to train system. We know many factors can be adjustors such as the central point or the shape of linguistic words in input or output fuzzy variables. The output variable affects outcome directly, so the central point is the adjuster in this system. The goal of genetic algorithm is to minimize the total error of each training data. The fitness of GA is to minimize the total error of every training item. *Min:* $E = ½(W_i\text{-}W_{obj})^2$ $i=1...n,$where n is number of training items.

2.2 Performance Analysis

2.2.1 Membership function and rules

The objective of the project is to develop an empirical model based manufacturing expert system, which is able to control the surface roughness of the precision metal cutting with changing process conditions. Two main factors feed-rate and tool-wear that are highly related to the surface roughness (Ra) of mechanical parts are studied. After a lot of experimental data is researched, the relationship of tool wear and Ra is high. In practice, tool wear is hard to measure; changed and by time and come with high noise while feed rate is relative easy to control and measured. Type-2 fuzzy system is very suitable to solve problem under the following conditions (N. N. M. Karnik, J.M., Qilian Liang, 1999). The triangle shape of the membership function will decrease the computational complexity in the software. The other shapes of the membership function which may be related to a priori knowledge will be studied in the future. Feed rate has seven membership functions. The row vector of it is: F= {VS, S, MS, M, MF, F, VF} where VS =very slow feed rate, S =slow, MS = medium slow, M = medium, MF =medium fast, F = fast, and VF = very fast. An isosceles triangle is used to represent the membership shape with fixed base length. It is shown in Fig. 4.

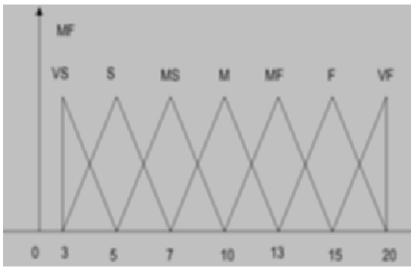


Fig. 4. Membership function of feed rate

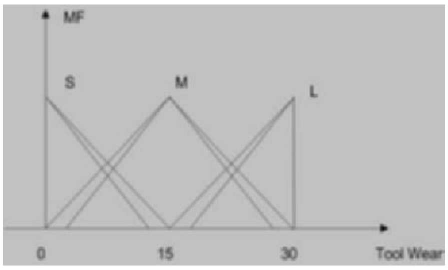


Fig. 5. Membership function plot of tool wear

Tool wear has three membership functions, such that TW= {S, M, L},where S = small, M = medium, L = large. The maximum allowable tool flank is set 0.030". Surface roughness has seven membership functions represented in singleton format. Ra= {VF, F, MF, M, MR, R, VR} ,where VF = very fine, F = fine, M = medium, MR= medium rough, R = rough, VR = very tough. The membership function figure is shown in Fig. 6.

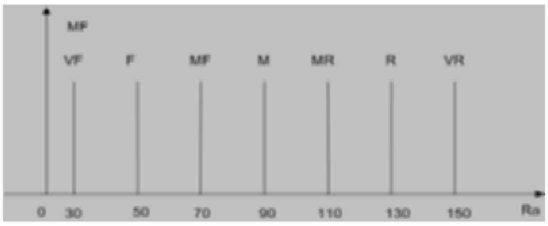


Fig 6. Membership functions plot for output variable Surface Roughness

How to draw appropriate fuzzy rules based on input and output membership functions for type-1 and type-2 fuzzy logic system is concentrated in this paper. For type-1 fuzzy logic system, since there are seven partitions for feed rate input variable and three partitions for tool wear input variable, there will be 21 rules.

TABLE I. Rules

		Feed Rate						
Ra		VS	S	MS	M	MF	F	VF
	S	VF	F	MF	MR	MR	R	R
Tool	M	MF	MR	MR	R	R	R	R
Wear	L	R	VR	VR	VR	VR	VR	VR

3 Results

3.1 Comparison of type-1 and hybrid Fuzzy Logic systems

Experimental data from Industry is shown on Fig. 7. The input variables are Feed Rate (*ipr*) and Tool Wear (*inch*) and the output variable is the Ra ($\mu \cdot inch$). In order to be shown clearly, we zoom in the data of Feed Rate x1000, and Tool wear x1000 on the Fig. 7. The history data provide the input to genetic algorithm training system. Since the size of training data is so small, we used cross-validation method as the size of subset is only 2. In fact, the incremental training method is better suitable for this application because of sequentially arriving data. However, the size of sample data set is too small, as result that we have to abandon incremental training method. We designed three cases type-1 without training, type-1 with training, and hybrid fuzzy system in order to highlight the performance of hybrid system. The output of three cases is shown on Fig. 8. In genetic algorithm training (John Koza, 1992), the parameters are chosen by population size=100 ;maximum generation=20;possibility value of Crossover =0.7;possibility value of mutation = 0.3 The training subsystem, which employs the empirical risk minimization (ERM) model, automatically modifies fuzzy logic membership function as system refinement. After training, we predict type-1 and hybrid fuzzy logic system separately. The error of three methods is listed on table II. The results show that hybrid Fuzzy logic system has a better result than type-1 Fuzzy logic system.

Since fuzzy rule and membership function depends on the expert's knowledge and experience, there exist some errors comparing to the real world. Then genetic Algorithm (GA) is employed to refine fuzzy membership function with experimental data. The results of predicted surface roughness Ra of type-1 fuzzy logic system with and without genetic algorithm training are much different. That tells us that training is necessary to fuzzy system. Otherwise, the system will not converge to the stable state until experts adjust the rules or membership function.

We can find in type-1 fuzzy logic system the predicting results of surface roughness Ra after training is more similar to experimental real data than the prediction results of surface roughness Ra without training. This indicates genetic algorithm training is successful to modify fuzzy membership function in type-1 Surface Roughness Control system. The results of predicted surface roughness Ra of hybrid fuzzy logic system before and after genetic algorithm training are also compared. No matter which method is used, the shape of curve is much closed to the real value, which showed us the rules base is defined in correct.

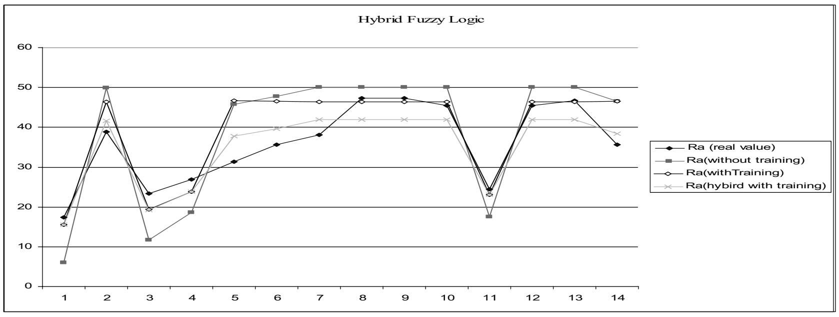


Fig. 7. Experimental data

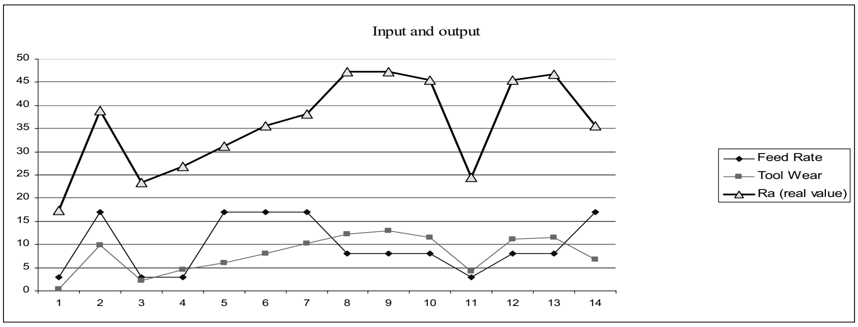


Fig. 8. Comparison of Ra of three methods

The results agree with our anticipation: the hybrid fuzzy logic systems, no matter whether it is trained or not, can achieve higher accuracy in predicting surface roughness than type-1 fuzzy logic system. The possible reason is that type-2 fuzzy logic system handle uncertainty in input variables more effective than type-1 fuzzy logic system.

TABLE II. The error of three methods of prediction

	Ra	Ra	Ra
	(without training)	(with training)	(hybrid with training)
Average Error	9.1996	6.7023	4.0278
Standard Deviation	61.9765	15.8856	11.6156

4 Conclusions

A surface roughness control system is established in this project. This control system is a fuzzy logic system to predict surface roughness Ra in manufacturing system in metal cutting. Since there is uncertainty in input fuzzy variables and fuzzy rules, two different fuzzy logic systems are applied in this system for comparison. One is type-1 fuzzy logic that uses Mamdani fuzzy method; another is type-1, type-2 hybrid fuzzy logic.

Genetic algorithm trained our surface roughness control system. Fuzzy rules are automatically modified in the process of genetic algorithm training. The results show that all system give out good results in our Surface Roughness Control system, specially, the hybrid fuzzy system demonstrated more precise prediction results of surface roughness Ra than type-1 fuzzy logic system. Genetic algorithm does improve the accuracy of system.

It is well-known that the contribution of different input variables to the output variable is different in the most control systems. There exist some primary input variables which play more important roles than others. The primary variables dominate the most part of output value. Currently, the genetic algorithm is used to tune the membership function of the output variable. This method which has directly tuned the consequents of rules may be more noise sensitive than tuning the antecedents since the antecedents decide consequents. Tuning the input variables is reasonable, especially, in those primary input variables. The advantage of the genetic algorithm compared to other algorithms is that it is highly domain independent. However, the genetic algorithm is not guaranteed to return the best result within the searching space (John Koza, 1992). The future works includes:

1.Find out the relationship between input variable and output variable via data mining technique such as rough set and association rule, thereby, find out primary input variables.

2.Adjust the membership function of primary input variables with GA or the new methods, such as neural network, which take advantage of the domain knowledge of the application.

At last, authors wish to thank Dr. Bill Tseng for his providing the experiment data and inspiring discussion.

5 References

J. C. C. Luke Huang, "A Multiple Regression Model to Predict In-process Surface Roughness in Turning Operation Via Accelerometer," vol. 17, 2001.

J. C. C. C. M. L. Mike S. Lou, "Surface Roughness Prediction Technique For CNC End-Milling," *Journal of Industrial Technology*, vol. 15, 1999.

J. M. Mendel, *Uncertain rule-based fuzzy logic systems : introduction and new directions*. Upper Saddle River, NJ: Prentice Hall PTR, 2001.

J. M. Qilian Liang, Mendel, "Interval type-2 fuzzy logic systems: theory and design," *IEEE Transactions on Fuzzy Systems*, vol. 8, pp. 535-550, 2000.

N. N. K., J. M. Mendel, "Operations on type-2 fuzzy sets," *Fuzzy sets and system*, 2001.

N. N. M. Karnik, J.M., Qilian Liang, "Type-2 fuzzy logic systems," *IEEE Transactions on Fuzzy Systems*, vol. 7, pp. 643-658, Dec 1999.

Reza Langari, "Fuzzy logic applications to control engineering," *SPIE Proceedings*, vol. 2061, pp. 2-7, Dec, 1993.

S. A. S. Coker, Y.C., "In-process Control of Surface Roughness Due to Tool Wear using a New Ultrasonic System,," *International Journal Machine Tools Manufacturing*, vol. 36, pp. 411, 1996.

S. Kalpakjian, "Manufacturing Processes for Engineering Materials, 3rd edition," *Addision Wesley*, 1997.

T. Hessburg, "Automatic design of fuzzy systems using genetic algorithms and its applications to lateral vehicle guidance," *SPIE Proceedings*, vol. 2061, pp. 452-463, Dec, 1993.

John R. Koza, " Genetic Programming: On the Programming of Computers by Means of Natural Selection" *MIT Press*, 1992

John R. Koza, "Survey of genetic algorithms and genetic programming". Proceedings of 1995 WESCON Conference. Piscataway, NJ: IEEE. 589 – 594,1995e.

Comparison of ANN and MARS in Prediction of Property of Steel Strips

Ananya Mukhopadhyay[†] and Asif Iqbal*

[†]R & D Division Tata Steel, Jamshedpur – 831 001, INDIA
Email: ananyamukhopadhyay@yahoo.co.in

*Pioneer Computers (P) Ltd. Jamshedpur -831 001, INDIA
Email: asifiqbal_jsr@hotmail.com

Abstract. Soft Computing has become popular in Steel Industry for its applications in the areas of reduction in defects, prediction of properties, classification of the products and many others. In recent times, the prediction of properties of steel strip is an area of increased interest mainly because of its prospective benefits of reduction in testing cost, better control on properties, reduction of inventory, increase in yield, and improvement in delivery compliance. Prediction of mechanical properties is a complicated task, as it depends on the chemical composition of the steel, and a number of processing parameters. In general, a high degree of nonlinearity exists between the property and the factors influencing it. In the past only Artificial Neural Network (ANN) was used, sometimes along with the variable reduction technique such as principle components / factor analysis. However, Multivariate Adaptive Regression Splines (MARS) has never been used despite some of its known advantages over the ANN. In this work two predictive models have been developed - one based on ANN, and another, MARS. This paper discusses on the model development and the comparative performance analysis of these two. The analysis shows that the results from both the models are comparable. However, shorter training time and automatic selection of important predictor variables, give MARS an edge over ANN.

Keywords: Soft Computing, Data Mining, MARS, ANN, Steel, Property Prediction

1. Introduction

Soft Computing is increasingly used in the Steel Industry particularly in the area involving finishing processes such as Hot Rolling (Elsilä and Röning 2002) and (Schlang et al. 2003). Hot rolled strips / coils are produced from the Hot Strip Mill (HSM) of a Steel Plant. The process involves prolonged heating of the slab in a furnace at elevated temperature (~1200 ºC), reduction of its thickness in a set of rolling stands by successive passes, followed by water-cooling down to 600-700 ºC, when it is coiled as the finished product. The various components of the HSM are shown in Figure 1. While rolling provides the desired final dimensions of the strip, water-cooling after rolling provides the required strength through metallurgical transformations. The finished coil is tested mechanically for conformance of mechanical properties - Yield Strength (YS), Ultimate Tensile Strength (UTS), and Elongation (EL), as specified by the customer in the order. The property obtained from the mechanical testing is posted in the test certificate of the coil before dispatch. The conventional mechanical testing procedure involves cutting of a sample from the outer wrap of the finished coil, machining to prepare the specimen of appropriate dimension followed

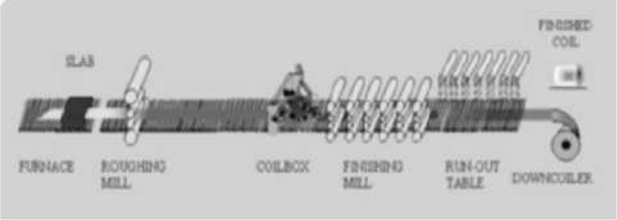


Figure 1. Schematic diagram of Hot Strip Mill

by tensile testing on a servo-hydraulic machine. Usually it is not before 24 hours that the test certificate is ready for any coil to be dispatched. This is a significant delay compared with the time taken to process a coil, which is about 7 minutes from furnace drop-out as slab to the final coiling as strip. This causes high inventory, delayed customer delivery, and high lock-in capital. Therefore, there has been a need to bring down the post-manufacturing time drastically. This motivates the development of a soft model to predict mechanical properties based on the steel-chemistry and the other actual rolling parameters for any given coil.

Such a soft computing based ANN model has indeed been developed for this purpose at the Hot Strip Mill of Tata Steel, India. Then, a full-fledged system known as the *'On-line Property PREdiction SyStem for Hot Rolled Coil (OPPRESS)'* has been developed in-house around the model (Innovation Intelligence, 2004). The system is in operation at our HSM, as shown in Figure 2. It predicts the mechanical properties of the strip in real-time, i.e., as soon as the coil is rolled. At present the system is available for low carbon unalloyed grade steel. The heart of this system is the prediction module based on a hybrid formulation using a combined Mathematical and ANN based model. The mathematical model is used to derive the cooling rate and the ferrite grain size, both of which are used as input to the ANN model. The other inputs to the ANN model come directly from the mill automation and information system. The mathematical model, which is based on the metallurgical knowledge, is also required to predict the mechanical properties of any new grades, which has never been rolled before, for which past data is not available. On the other hand, the ANN model plays a significant role for prediction of properties of the ongoing grades. The system has helped reduce mechanical testing of cold rolling quality grade coils drastically. The cold rolled commercial quality coils are first produced in the Hot Strip Mill as hot rolled coils, to be further processed in the Cold Rolling Mill of the steel plant. These cold rolling quality grades form a major part of the product mix of the Hot Strip Mill.

Figure 2. OPPRESS system in service to HSM, Tata Steel, India

The model used in the OPPRESS system is an on-line model. The prerequisite of any on-line model is the development of an off-line model. Such ANN based off-line models for the prediction of properties was also developed by other authors for carbon and niobium microalloyed grade steels (Dumortier et al. 1998). In this work a statistical model was developed using the Principal Component Analysis to reduce the number of variables, which were then given as input to the model. A feed-forward network of 17-13-7-1 topology was used. As there was only one output (YS/UTS/EL), three separate models were trained, one each for YS, UTS, and EL. In their work, however, optimization of network topology was not done. But, it was found from their study that the application of ANN gave better results compared with the Multiple Linear Regression. Guodong et al. (2000) used two ANN models, one for prediction of strength and another for elongation. A 6-10-1 network architecture was taken for the prediction of strength and elongation separately for C-Mn steel. Döll et al. (1999) developed a mathematical, and an ANN-based hybrid model for prediction of mechanical properties of low-carbon manganese steels. The chemistry and process parameters were first fed through the mathematical model to obtain the ferrite grain size. This grain size along with seven other process parameters was then used as input to the ANN model. Accuracy levels of ± 30 MPa for YS, and ± 20 MPa for UTS were achieved. The Root Mean Squared (RMS) error was obtained as 11 MPa and 8 MPa for YS and UTS respectively. Korczak et al. (1998) used a neural network model with the 14-7-4 architecture to predict ferrite grain size, hardness, YS and UTS for low carbon hot rolled steel plates. The results obtained from the ANN model were compared with the experimental measurements.

The accuracy of the ANN model in the existing OPPRESS system at Tata Steel's HSM is ± 15 MPa for YS, and ± 15 MPa for UTS, and ± 5% for EL. Although such level of accuracy is acceptable for industrial application, an attempt has been made to improvise it even further in terms of performance or otherwise. While application of ANN exists in the literature, no attempt has so far been made, as per authors' knowledge, to explore other techniques such as MARS.

In the present work, an attempt has been made to develop a MARS model and compare its predictive capability with the already developed ANN model used in the OPPRESS system. The same data set, used for the development of ANN, was also used for development of MARS. During development of the ANN model different topologies were used to build different network configurations, and the best network was chosen based on the predictive capability. Similarly in case of MARS, different model configurations were tried out before selecting the best. The best selected ANN and MARS models were then applied on a same set of unknown records for prediction. The results obtained were then compared. This paper describes a comparative analysis of the performance of the two predictive models.

2. Method

2.1 Data

A brief description of the data set preparation for the development of the models is given below.

The OPPRESS is an on-line system. Hence, it uses the on-line ANN model, which is the expression of YS, UTS, and EL in terms of weights and the input variables. These weights are previously determined from an off-line ANN model, developed through training, selection, and test procedure. Data used to train this off-line model was the historical data taken from the Data Warehouse. The Hot Strip Mill of Tata Steel is equipped with the state-of-the-art Data Warehouse where the production data related to each coil is stored. The Data Warehouse is based on IBM DB2 UDB database residing in the RS6000 server machine of 144 GB Hard disk. The steel chemistry, the operational data, and the actual mechanical testing data for all the coils are also stored in the Data Warehouse.

From literature the seven variables responsible for imparting the strength of steel were identified. These are – 1) Strip thickness, 2) Carbon Equivalent, 3) Cooling Rate, 4) Ferrite Grain Size, 5) Rolling speed, 6) Finish Rolling Temperature, and 7) Coiling Temperature. These predictor variables are named by X1… X7 respectively (Table 1). And the response variables are named by YS (X8), UTS (X9), and EL (X10). The data, comprising seven predictor variables and three response variables, for a large number of coils were extracted from the Data Warehouse through SQL (Structured Query Language) query. A total of 1209 records were collected from the database. These data were used for the purpose of Data Mining. The Mining software, (STATISTICA™ DATA MINER 1984-2003), used for this work, resides in IBM PC with Pentium III processor of 866 MHz, 128 MB RAM, and 19 GB Hard disk.

Table 1. Statistical distribution of data for different variables used in ANN and MARS model.

Var_id	Variable Description	Unit	Mean	Std. Dev	Min	Max
X1	Strip Thickness	mm	2.74	0.71	1.80	4.00
X2	Carbon Equivalent (10^{-2})	% wt	7.86	2.14	4.62	14.00
X3	Cooling Rate	°Cs^{-1}	57.49	15.04	23.87	99.98
X4	Ferrite Grain Size	μm	15.60	1.48	10.00	18.75
X5	Rolling Speed	ms^{-1}	10.44	1.80	6.25	13.00
X6	Finish Rolling Temperature	°C	894.42	12.14	861.00	920.00
X7	Coiling Temperature	°C	572.21	20.06	520.00	640.00
X8	Yield Strength	MPa	285.43	23.19	222.00	340.00
X9	Ultimate Tensile Strength	MPa	359.92	20.88	321.00	400.00
X10	Elongation	%	40.04	3.09	34.00	46.00

2.2 Predictive Models

The objective of the development of the predictive model was to express the three response variables in terms of the seven (maximum) predictor variables. The requirement of the model was that it should be accurate and reliable enough to be used as a substitute for mechanical testing.

The Learn-and-Recall type predictive models can be placed in a scale covering from local to global models based on the learning philosophy (Dwinnell 2000). A local model is an agglomeration of several "micro-models" which operate locally in the input space, thus can represent the local features sufficiently well. Typically they are fast to train but slow to recall. On the other hand, a global model employs the philosophy of creating a single giant model covering the features of the entire input space. Thus global models are slow to train, but quick to recall. MARS works under the philosophy of dividing the input space into several sub-regions and fitting splines to these local regions to build the final model. Therefore, it is more close to the local end of the spectrum. On the contrary, a feed-forward ANN works under the principle of developing a complete model that covers the space more globally between the input and the output. Hence, it falls more towards global end of the spectrum. The suitability of any model by and large depends on the data, the relationships among the variables, and their complexities.

The development of the off-line ANN model for the OPPRESS system, and the present MARS models, is described below.

2.2.1 ANN Model Development

ANN is a system of interconnected nodes and weights, used for processing information in the same way as does the biological nervous system. Multi-layer Perceptron (MLP) is a type of ANN where there is an array of neuron or processing unit at the input layer, and an array of neurons in the output layer. In between there are many hidden layers. Each neuron in the input layer is connected to all other neurons in the next hidden layer. And each neuron in the hidden layer is connected to all other neurons in the next hidden layer, and so on, until the connectivity is established to the output layer as shown in Figure 3. In feed-forward network, information flows only in one direction - from input to output. The performance of the network depends on the weights of the connection and the transfer function used for the neurons. The model development procedure involves three steps.

a) the network is presented with training examples, which consist of a pattern of activities for the input units together with the derived pattern of activities for the output units.

b) how closely the actual output of the network matches with the derived network, is determined.

c) the weight of each connection is changed so that the network produces a better approximation of the desired output.

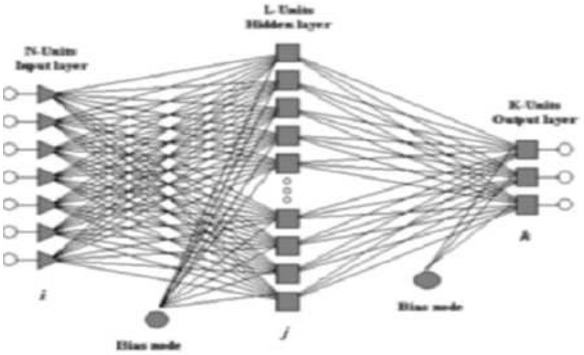


Figure 3. Artificial Neural Network showing connectivity of neurons.

In case of ANN, the model was trained with 967 records and validated (selection) with another 121 records. The remaining 121 records out of the total of 1209 records were used for testing (prediction). This constitutes a 80/10/10 split of the total data set. The data partitioning was done at random. The feed-forward ANN model was developed with several different network topologies (Mukhopadhyay and Iqbal 2004). Out of these the best three network topologies are presented here in Table 2. One and two hidden layers were considered with varying number of neurons. Also, three neurons in the output layer were used for the three response variables, and seven neurons were used in the input layer for the seven predictor variables. The values of the variables in the input data set were normalized using min-max function before they were fed as input to the network. A sigmoidal transfer function was considered. In the learning phase, the Back-propagation (BP) algorithm was chosen to train the network. The learning rate and the momentum rate were chosen appropriately for the training run. However, to accelerate the training process Conjugate Gradient (CG) Descent algorithm was also adopted after completion of 1000 epochs. The stopping condition was given on the basis of the number of epochs.

For each network, the Normalised Mean Squared Error (NMSE) was calculated. It is the ratio of variance of prediction error over the actual variance of the desired data. This is given as

$$\text{NMSE}=\frac{\sum_{p=1}^{P}\left(O_{pk}-t_{pk}\right)^2}{\sum_{p=1}^{P}\left(t_{pk}-\bar{t}_{pk}\right)^2} \qquad (1)$$

where, O_{pk} is the actual output, t_{pk} is the target or desired output, $\bar{t}_{pk}$ is the mean of the target output for the p-th pattern at the k-th neuron in the output layer. P is the total number of patterns or records. The Standard Deviation Ratio (S. D. Ratio) and the coefficient of determination (R^2) were calculated using the following expressions.

$$\text{S. D. Ratio} = \sqrt{NMSE} \qquad (2)$$

R^2=1–

NMSE (3)

Sensitivity analysis was carried out to understand the sensitivity of the developed ANN model with respect to the input variables. To obtain the relative importance of these input parameters on the sensitivity of the model, a ranking method was used. To determine the relative sensitivity of any given variable, the values of the variable in the training dataset were all replaced by its mean value keeping the magnitudes of other variables unchanged. The neural network model was then run to accumulate the network error. The ratio of the accumulated network error obtained from the original dataset during training and that obtained from the substituted dataset for a given variable was calculated. If the variable is very sensitive to the network model then the error deteriorates greatly and, therefore, the ratio becomes large. If the ratio is greater than one, it means that the variable is important to the network model. If the ratio is one or less than one, it means that the variable has no significant effect on the performance of the network. When the ratios of the relative sensitivity of all variables were calculated, the ranking of individual variables were done in the decreasing order of the ratios.

2.2.2 MARS Model Development

Multivariate Adaptive Regression Splines (MARS) is an advanced non-parametric, non-linear technique developed by (Friedman 1991). The space between the input and the output is represented through localized adjustments using splines. The model is of the form of an expansion of piecewise linear functions. It is used for both prediction (in case of continuous variable) and classification (in case of categorical variable) problems. MARS can handle missing values in the training sample. It works better when interaction involves a few variables and the model size is small (Weiss and Indurkhya 1998). A detailed description may be obtained in (Duarte et al. 2004). The model development procedure involves three steps.

a) a method to select local spline functions and to partition the original data domain.

b) a backward procedure to prune unnecessary splits to achieve the minimum number of spline functions.

c) a smoothing procedure to provide the final model with a certain degree of continuity at the knots.

The model is of the form of an expansion of piecewise linear functions, and is given by

$$\hat{y} = \sum_{m=1}^{M} a_m . B_m(X) \tag{4}$$

$$B_m(X) = \prod_{k=1}^{K_m} H\left[s_{k,m}\left(X_{v(k,m)} - T_{k,m}\right)\right] \tag{5}$$

where $\hat{y}$ is the predicted value, M is the number of basis functions, a_m is the local adjustment, $B_m(X)$ is the Basis Function(BF), and X is the vector of independent variables. $H[.]$ is the step function, K_m is the number of domain splits, $s_{k,m}$ is ± 1, $T_{k,m}$ is the knot of the splits (Figure 4), and $X_{v(k,m)}$ is the vector of independent variables.

In this case, the MARS model was developed with the same set of 1088 records that was earlier used for ANN (training and selection together). In order to find the best MARS model, three models were developed – the first one was additive and the remaining two were interactive with two and three variables interactions. The penalty and the threshold values used for all cases were kept same as 2 and 0.0005 respectively. For all the models the Generalized Cross Validation (GCV) error was calculated to measure the goodness-of-fit.

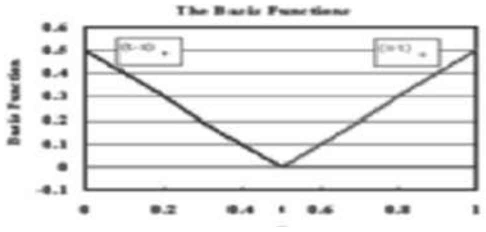


Figure 4. Schematic of the basis functions used in MARS

$$\text{GCV(M)} = \frac{1}{P} \cdot \frac{\sum_{i=1}^{P} \left(y_i - \hat{y}(X_i)\right)^2}{\left(1 - \frac{C(M)}{P}\right)^2} \tag{6}$$

where y_i is the actual output value for any given observation, P is the number of observations. The term, $\frac{1}{\left[1 - \frac{C(M)}{P}\right]^2}$ is the penalty for using M basis functions. *C(M)* was calculated using the following relationship.

$$C(M) = C_1(M) + d \tag{7}$$

$$C_1(M) = \text{trace}\left(B\left(B^T B\right)^{-1} B^T \right) + 1.0 \tag{8}$$

d was taken as 2, according to (Hastie et al. 2001). *C(M)* depends on the number of basis functions employed.

The residual was calculated for all the three models. The adjusted R^2 value was calculated using

$$Adj\ R^2 = 1 - \left(1 - R^2\right) \times \frac{(P-1)}{(P-k-1)} \tag{9}$$

where R^2 is coefficient of determination, P is number of records, and k is number of independent variables.

3. Results and Discussion

Table 1 shows the descriptive statistics for all the seven predictor variables (X1,..,X7), and three response variables (X8,..,X10) for a total number of 1209 observations used for Data Mining in ANN and MARS. The variables were found to be nearly normally distributed. The mean, standard deviation, minimum, and maximum values for each variable are also shown in the table.

In case of ANN, the training times for the three ANN configurations were found to be as – 39 seconds with 1000 epochs of BP and 107 epochs of CG for 7-3-3 network; 18 minutes and 34 seconds with 1000 epochs of BP and 242 epochs of CG for 7-19-3 network; and 28 minutes and 11 seconds with 1000 epochs of BP and 2462 epochs of CG for 7-22-3 network. Table 2 shows the mean and the standard deviation of error during selection (validation) stage, resulting from the difference between the actual and the predicted values of all the three response variables. The standard deviation ratio and the coefficient of determination were calculated using using Equations (1), (2) and (3) and are shown in the table. It can be seen that the error mean is minimum for all three response variables in case of 7-19-3 compared with 7-3-3 or 7-22-3 network. Also, the error standard deviation for 7-19-3 is lower compared with the other two, except in case of X10 where error standard deviation is slightly higher than that for 7-3-3 and 7-22-3. The S.D ratio is lower for 7-19-3 compared with the other two networks except in case of X10 for 7-22-3. On the other hand, R^2 value for 7-19-3 is the highest among these three for all response variables. Out of the three network configurations, the 7-19-3 network was chosen as the best one (shown with asterisk in Table 2) based on minimum error and maximum R^2.

The 7-19-3 network was then applied to 121 unknown records where only values of the predictor variables (no response variables) were given to the network to test (or predict). The predicted values of the all three response variables plotted against their actual values. These are shown in Figures 5(a)-(c). Table 5 shows the different error statistics for prediction. The mean error and its standard deviation are shown in the table along with the coefficient of determination.

Table 2. Selection of the best ANN Model

Network Topology	**Error Mean**			**Error S.D.**		
	X8	**X9**	**X10**	**X8**	**X9**	**X10**
7-3-3	-0.79	-0.83	0.25	13.60	11.00	2.15
*7-19-3	-0.25	-0.54	0.11	12.21	10.49	2.17
7-22-3	0.99	0.79	0.22	12.78	10.68	1.93
	S.D.Ratio			**R^2**		
	X8	**X9**	**X10**	**X8**	**X9**	**X10**
	0.58	0.55	0.70	0.672	0.706	0.504
	0.51	0.49	0.69	0.757	0.757	0.518
	0.56	0.51	0.64	0.689	0.740	0.593

Table 3 shows the ratio of errors and the sensitivity ranking of the input variables based on the select data set. The rank is obtained on the basis of the descending order of the error ratio. Thus it can be said that for low carbon unalloyed steel, the strip thickness, carbon equivalent, and ferrite grain size are the three most influential parameters for the prediction of mechanical properties. It can be noted that the effect of cooling rate on mechanical property is not significant.

Table 3. Ranking of variables based on sensitivity

Sl No.	**Variable**	**Error Ratio**	**Rank**
1	Strip Thickness	1.375249	1
2	Carbon Equivalent	1.283869	2
3	Cooling Rate	1.026432	5
4	Ferrite Grain Size	1.113477	3
5	Rolling Speed	1.048669	4
6	Finish-Rolling Temperature	1.004485	6
7	Coiling Temperature	1.002554	7

In case of MARS, the time duration for development of MARS1 is 4 seconds, for MARS2 9 seconds, and for MARS3 13 seconds. This time duration is significantly less compared with the ones reported earlier in case of ANN.

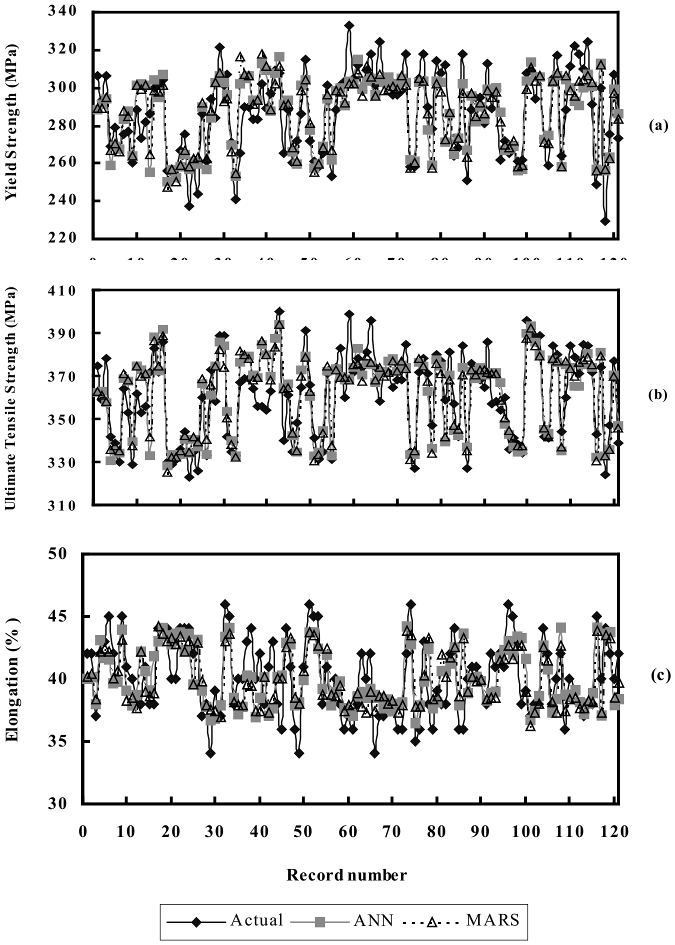


Figure 5. Comparison between predicted and actual values for (a) YS, (b) UTS, and (c) EL

Table 4. Selection of the best MARS model

Model	DI	NB	Adjusted R2			Residual			GCV error
			X8	X9	X10	X8	X9	X10	
						σ	σ	σ	
MARS1	1	11	0.675	0.738	0.519	13.20	10.67	2.14	0.2804
*MARS2	2	23	0.694	0.745	0.522	12.78	10.50	2.13	0.2694
MARS3	3	22	0.691	0.743	0.519	12.87	10.54	2.14	0.2706

DI-degrees if interaction; NB-Number of basis functions; μ for Residual is zero

Table 5. A performance Comparison of best ANN model (7-19-3) and best MARS model (MARS2) during Prediction.

Model	Error Mean			Error S. D.			R^2		
	X8	X9	X10	X8	X9	X10	X8	X9	X10
7-19-3	-0.29	0.49	-0.13	12.92	10.62	2.25	0.661	0.727	0.454
MARS2	0.14	-0.66	0.21	12.93	10.44	2.25	0.662	0.735	0.456

The number of basis functions finally used after pruning for all three MARS models are shown in the Table 4. During training stage the model generates many basis functions for mapping the space between the input and the output. During pruning the model discards unnecessary basis functions based on the cross-validation error. In case of additive model, MARS1, the least number of basis functions were used compared with two or three variable interactions in MARS2 and MARS3 respectively. Equation (10) shows that a total of 23 basis functions were selected for MARS2 model. The basis functions, BF1 - BF14 are for one interactions and BF15 - BF23 are for two interactions.

BF1 = MAX (0, X2 -0.069)
BF2 = MAX (0, 0.069 - X2)
BF3 = MAX (0, X1 - 2.8)
BF4 = MAX (0, 2.8 - X1)
BF5 = MAX (0, 2.6 - X1)
BF6 = MAX (0, X6 - 877)
BF7 = MAX (0, X4-17.83)
BF8 = MAX (0, 17.83 - X4)
BF9 = MAX (0, X5 - 12.98)
BF10 = MAX (0, 596 - X7)
BF11 = MAX (0, X6 - 917)
BF12 = MAX (0, 917 - X6)
BF13 = MAX (0, X4 -16.16)
BF14 = MAX (0, X2 -0.082)

BF15 = BF5 * BF1
BF16 = BF3 * BF7
BF17 = BF3 * BF8
BF18 = BF9 * BF6
BF19 = BF4 * BF10
BF20 = BF4 * BF11
BF21=BF4*BF12
BF22 = BF1 * BF13
BF23= BF4 * BF14 (10)

MARS also automatically identifies the "knot" for the functions (as shown for BF1 through BF14) from the input data. It is interesting to note that there exists no basis function involving the variable X3, i.e., the cooling rate. Thus the automatic feature selection procedure of MARS excluded the cooling rate as of little importance. The sensitivity of the ANN model shown in Table 4 also shows a low rank of the cooling rate for the type of steel considered. Table 5 shows the performance of three MARS models in terms of Adjusted R^2, Residual, and GCV error. Based on the lowest residual and GCV error, and highest adjusted R^2, the MARS2 model has been considered as the best (showed with asterisk).

The YS, UTS and EL expression in terms of the 23 basis functions for MARS model are given in equations (11) – (13).

$$\begin{aligned} YS = {} & 286.11 + 130.91*BF1 - 1184.04*BF2 - 9.85*BF3 + 40.59*BF4 -1332.16* \\ & - 0.31*BF6 + 15.61*BF16 + 4.31*BF17 + 24.88*BF18 + 0.21*BF19 \\ & - 0.21*BF19 - 5.50 *BF20 - 0.37*BF21 - 2976.75 * BF22 + 127.36*BF23 \end{aligned} \tag{11}$$

$$\begin{aligned} UTS = {} & 358.54 + 421.83*BF1 - 1254.54*BF2 - 4.92*BF3 + 23.81*BF4 - 15.27*BF15 \\ & - 0.16*BF6 + 15.42*BF16 + 2.40*BF17 + 10.94*BF18 + 0.16*BF19 -07* \\ & BF20-0.32*BF21-1470.98*BF22+327.88*BF23 \end{aligned} \tag{12}$$

$$\begin{aligned} EL = {} & 41.47 - 38.14*BF1 + 106.33*BF2 - 0.25*BF3 - 7.07*BF4 + 241.04*BF15 \\ & + 0.016*BF6 - 1.27*BF16 - 0.08*BF17 - 0.35*BF18 + 0.003*BF19 - 0.32* \\ & BF20 + 0.023*BF21+138.89*BF22-145.27*BF23 \end{aligned} \tag{13}$$

The MARS2 model was then applied to the same set of 121 unknown records that was used in ANN. The predicted values were plotted against the actual values. These are shown in Figures. 5(a)-(c) for comparison. From Figures. 5(a)-(c) it can be seen that the prediction performance of both the models are comparable. Table 6 shows the statistical comparison of prediction performance between the best ANN model (7-19-3) and the best MARS model (MARS2), obtained from Figures. 5(a)-(c). It shows that no significant difference exists between the ANN and MARS model as far as the prediction accuracy is concerned.

Coming back to the industrial problem of improvising the performance of the existing OPPRESS system at Tata Steel's HSM, MARS and ANN have been compared. Although the improvement of results cannot be expected from the replacement of ANN model with MARS, as has been seen here, there are other benefits. The difficulty of any on-line system is that the model requires to be adapted or fine-tuned at a regular interval of time to catch up with the Plant's natural operational drift. This means that the weights of the existing ANN model require adjustments over time. The time for periodic revalidation of the existing on-line ANN based model can be reduced drastically with the replacement of ANN model with MARS.

4. Conclusion

Thus it has been found that no significant difference in performance exists between the ANN model and the MARS in the application of property prediction. However, MARS model is easier to build. Also, it automatically takes care of unimportant and redundant predictor variables. Compared with ANN, it also takes much less time totrain. From the point of view of industrial applicability an on-line model is required to be adapted at a regular interval of time to take into account the natural operational drift of the mill. This adaptation becomes much easier and quicker with MARS than ANN. Therefore, it has been decided that the existing ANN model of OPPRESS system will be replaced by MARS in near future.

Acknowledgements

We would like to thank the management of Tata Steel, India for kind permission to publish this work.

References

Döll, R., Sorgel, R., Daum, M., and Zouhar, G. (1999), "Control of mechanical properties," *Asia Steel*, Crambeth Allen Publishing, Craven Arms, UK.

Duarte, B., Saraiva, P. M., and Pantelides, C. C. (2004), "Combined Mechanistic and Empirical Modelling," *International Journal of Chemical Reactor Engineering*, vol. 2, pp. 1-19.

Dumortier, C., Lehert, P., Krupa, P., and Charlier, A. (1998), "Materials Science Forum," vols. Trans. Tech Publications, Switzerland, pp. 284-393.

Dwinnell, W. (2000), "Exploring MARS: An Alternative to Neural Network," *PC AI Magazine*.

Elsilä, U., and Röning, J. (2002), "Knowledge Discovery in Steel Industry Measurements," *Proc of Starting Artificial Intelligence Researchers Symposium (STAIRS)*, Lyon, France, pp. 197- 206.

Friedman, J. H. (1991), "Multivariate Adaptive Regression Splines," *The Annals of Statistics*, vol. 19, pp. 1-141.

Guodong, W., Xianghua, L., Cheng, L., Xiumei, W., and Tong, W. (September 26-29, 2000), "Application of AI in Rolling and Prediction of Properties of Hot Rolled Materials", *Asia Steel International Conference* (Rolling), Beijing, China, pp. 38-47.

Hastie, T., Tibshirani, R., Friedman, J. (2001), "The Elements of Statistical Learning: Data Mining, Inference and Prediction," Springer-Verlag, N.Y, USA.

Invention Intelligence: S&T Magazine, New Delhi, India, (May-June 2004), vol. 39, no. 3, pp. 125-126.

Korczak, P., Dyja, H., and Łabuda, E. (1998), "Using Neural Network Models for predicting Mechanical Properties after Hot Plate Roll Process," *Journal of Materials Processing Technology*, vol. 80-81, pp. 481-486.

Mukhopadhyay, A., and Iqbal, A. (2004), "Prediction of Mechanical Properties of Hot Rolled, Low-Carbon Steel Strips using Artificial Neural Network," *Materials and Manufacturing Processes, (2005) vol. 20,no. 5, pp. 739-746*

Schlang, M., Feldkeller, B., Lang, B., Poppe, T., and Runkler, T. (2003), "Neural Computation in Steel Industry," *European Summer School on Multi-Agent Control*, Hamilton Institute, National University of Ireland, Maynooth, Ireland.

STATISTICA™ DATA MINER ver 6.1: StatSoft, Inc., Tulsa, OK 74104, USA, (1984-2003).

Weiss, S. M., and Indurkhya, N. (1998), *Predictive Data Mining – A Practical Guide*, Morgan Kaufmann Publishers, Inc., San Francisco.

Designing Steps and Simulation Results of a Pulse Classification System for the Electro Chemical Discharge Machining (ECDM) Process – An Artificial Neural Network Approach

T.K.K.R. Mediliyegedara[1], A.K.M. De Silva[1], D.K. Harrison[1], J.A. McGeough[2], D.Hepburn[1]

[1] School of Engineering, Science and Design, Glasgow Caledonian University, Glasgow, U.K.

[2] School of Engineering and Electronics, The University of Edinburgh, Edinburgh, U.K.

Abstract. This paper presents the designing steps and simulation results of a pulse classification system for the ECDM process using artificial neural networks (ANN). An Electro Discharge Machining (EDM) machine was modified by incorporating an electrolyte system and by modifying the control system. Gap voltage and working current waveforms were obtained. By observing the waveforms, pulses were classified into five groups. A feed forward neural network was trained to classify pulses. Various neural network architectures were considered by changing the number of neurons in the hidden layer. The trained neural networks were simulated. A quantitative analysis was performed to evaluate various neural network architectures.

Keywords: ECDM, Artificial Neural Networks, Pulse Classification, ECDM process Control

1 Introduction

Electro Chemical Discharge Machining (ECDM) is a hybrid non-conventional manufacturing process which combines the features of Electro Chemical Machining (ECM) and Electro Discharge Machining (EDM) (Mediliyegedara et al. 2004). The ECDM process consists of a cathodic tool and an anodic workpiece, which are separated by a gap filled with electrolyte, and pulsed DC power applied between them. This leads to electrical discharges between the electrodes, thus achieving both electrochemical dissolution and electro-discharge erosion of the workpiece (De Silva et al. 1995). One of the major advantages of ECDM, over ECM or EDM, is that the combined metal removal mechanisms in ECDM, yields a much higher machining rate (De Silva, 1988).

The performance of ECDM, in terms of surface finish and rate of machining, is affected by many factors. Relationships between these factors and machining performance are highly non linear and complex in nature. Therefore, it is very difficult to develop a relationship between those factors and the machining performance with conventional mathematical modelling. This fact makes it very hard to formulate control strategies for the process control of ECDM (Mediliyegedara et al. 2004). Pulse classification plays a vital role in the formulation of control strategies. Strategies for pulse classification in EDM have been studied in the past but there remains a need for an effective and efficient pulse classification system for ECDM.

Tasi and Wang (2001) have utilised ANNs to model the metal removal rate in electro-discharge machining. Both Liu and Tarng (1997) and Kao and Tarng (1997) employed feed-forward neural networks for the on-line recognition of pulse types in the EDM process. Based on their results, discharge pulses were identified and then employed for controlling the EDM process. Pajak and Wieczorowski (1998) have employed unidirectional multilayer neural networks for the classification of discharges in Electro Contact Discharge Machining. They classified these electrical discharges into three groups such as "simple electric discharges", "multiple electric discharges" and "continuous electrical discharges". Mean current intensity and value of the amplitude harmonic spectrum of the current intensity were utilised as inputs for the neural network.

Without an efficient pulse classification system, there are many drawbacks in the ECDM process. Firstly, a distortion of the workpiece surface can result due to overheating. This distortion leads to a poor surface quality. Secondly, the rate of metal removal will reduce due to the enlarging of the machining gap, because of any inefficiency in the pulse classification system, the control algorithm will make a wrong decision leading to a larger machining gap. Therefore, pulse classification plays a vital role in the formulation of control strategies. Thus, an intelligent pulse classification system for ECDM is a useful research approach to pursue.

2 Pulse Types in the ECDM Process

It is possible to identify five distinct types of pulses in the ECDM process such as Electro Chemical Pulse (ECP), Electro Chemical Discharge Pulse (ECDP), Spark

Pulse (SP), Arc Pulse (AP) and Short Circuit Pulse (SCP). SP and AP are also known as Normal Discharge Pulse and Abnormal Discharge Pulse respectively. Open Circuit Pulse (OCP) is not present in ECDM as some electrochemical current flows even with a larger gap (De Silva et al. 1995). De Silva (1988) has presented a detailed analysis of various pulses in ECDM.

3 Experimental Setup

An EDM machine was modified by incorporating an electrolyte system and changing the control system. $NaNO_3$ was used as the electrolyte (Figure 1). A mild steel work piece and copper electrode were used. The duty ratio and pulse duration were set to be 50% and 100µs respectively. The above mentioned five types of pulses were acquired using a storage oscilloscope at a sampling frequency of 1 MHz. The MATLAB 6 software package was used to model and to simulate the Pulse Classification System (PCS).

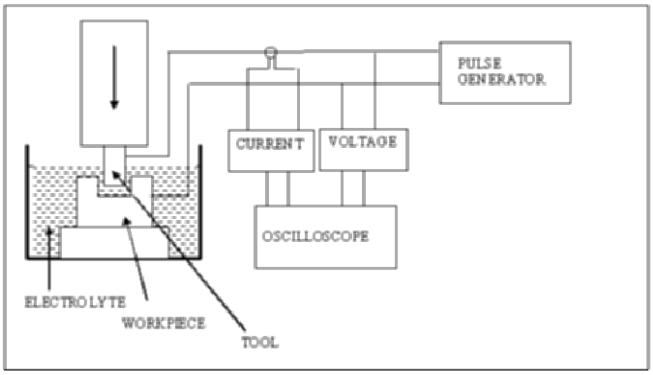


Figure 1: A schematic diagram of the experimental set-up

4 Designing of the Classification System

When an ANN is used for the pulse classification it necessary to identify the most appropriate neural network architecture. As far as real time implementation is concerned, there are many important parameters that must be investigated. Firstly, a suitable neural network architecture must be identified. Secondly, one has to identify the features that can be effectively used to classify pulses. Thirdly, it is necessary to prepare a suitable training data set and a test data set. Fourthly, the optimum number of layers and the number of neurons in the each layer has to be decided. Fifthly, it is necessary to investigate an activation function that is easy to implement, while providing acceptable classification accuracy. Finally, a training algorithm, which provides efficient training, has to be identified. In this particular application, training can be performed offline at the designing phase. Therefore this application does not demand an investigation of efficient training algorithms.

4.1 Neural Network Architecture

In the past, researches have found that the feed-forward neural network architecture will provide the better performance in the pulse classification of EDM process (Liu and Tarng, 1997 and Kao and Tarng, 1997). Therefore, It is decided to use a feed-forward ANN classify pulses in the ECDM process. One of the most widely used artificial neural networks is the feed-forward neural network architecture also known as Multi-Layered Perception (MLP) (Bermak and Bouzerdoum, 2002). The popularity of the MLP architecture stems from the existence of efficient training techniques based on the back-propagation algorithm. In a feed-forward architecture the information propagates from the input to the output in a feed-forward manner, passing through intermediate processing layers called hidden layers. A feed-forward architecture may contains one or more hidden layers. Each hidden layer comprises processing elements, or neurons that receive inputs only from the neurons in the preceding layer; there is no information flow between neurons residing in the same layer. The general architecture of the MLP neural network is shown in Figure 2.

4.2 Feature Extraction

Four different features were considered when classifying pulses such as Peak Voltage (PV), Average Voltage (AV), Peak Current (PC) and Average Current (AC). Since the four features are used as inputs, four neurons are used in the input layer such as $I_{1,}$ I_2, I_3 and I_4. Similarly, since there are five distinct types of pulses, five output neurons are used in the output layer such as O_1, O_2, O_3, O_4 and O_5.

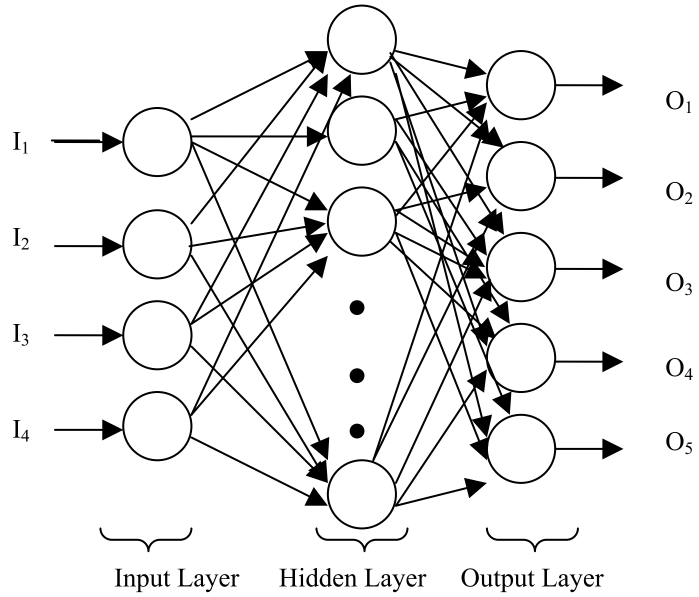


Figure 2: The general architecture of the MLP neural network

4.3 The Preparation of a Training Data Set and a Test Data Set

One hundred pulses were selected from each pulse type and the following values were calculated, Peak Voltage (PV), Average Voltage (AV), Peak Current (PC) and Average Current (AC). PV, AV, PC and AC were used as the features (inputs) in the ANN. Outputs of the ANN were prepared as follows. If a pulse belongs to ECP, '1' is assigned to the ECP and '0' is assigned to the other pulses. Similarly, 'ones' and 'zeros' are assigned to all the pulse types to prepare an output matrix. The data set was divided into two sets, the training data set and the test data set. The training data set and the test data set consist of 70 and 30 data points respectively for one type of pulses. Therefore, altogether the training data set and the test data set consist of 350 and 150 data points.

4.4 Number of Layers and Number of Neurons in Each Layer

In real time implementation point of view, the lesser the number of layers the lesser the calculation cycle time. Therefore a FFNN with a one hidden layer was considered in this study. There are four inputs in the input layer. PV, AV, PC and

AC were used as the inputs. There are five outputs such as ECP, ECDP, SP, AP and SCP. Now, one has to investigate the optimum number of neurons in the hidden layer and the best activation function having less complexity. Logistic Sigmoid Function (LOGSIG) activation function was used in each neuron.

5 Definition of Classification Accuracy

It is necessary to have a method to measure the performance of a PCS to investigate the most suitable neural network architecture. Classification Accuracy (CA) is introduced to compare the performance of the PCS One can define classification accuracy of the PCS as the average CA of each type of pulses. In general, the CA of a 'X' type pulse can be defined as follows.

$$\text{CA} = \left\{ \frac{\sum_{i=1}^{n_x} x_i}{n_x} - \frac{\sum_{i=1}^{n_y} y_i}{n_y} \right\} \times 100\ \% \qquad (1)$$

Where,

x_i - Simulated output value from 'X' output for *i*th pulse when the input values correspond to 'X ' type pulses,

y_i - Simulated output value from all other outputs for the *i*th pulse when the input
correspond to 'X' type pulses,

n_x - Number of 'X' type of pulses,

n_y - Number of all other type of pulses (Since there are 150 data points in test data set, $n_y = 150\text{-}n_x$)

6 Simulated Results

The simulated results, which are shown in the following Figures, are corresponding to a trained ANN having six neurons in the hidden layer (i.e. N=6). The test data set as outlined in section 4.3, consisted of 150 data points, 30 of each of the five types of pulse. Vertical axes (Y) of the following graphs indicate the output values of the neural network. In the ideal situation, if a pulse is an ECP, then the output value from node O_1 should be equal to '1'. Other output values (O_2, O_3, O_4 and O_5) should be equal to '0'.

Figure 3 shows the output O_1 is nearly equal to ‘1’, for the first 30 pulses. That means the first 30 pulses have been classified as ECP by the ANN. Similarly Figure 4, Figure 5, Figure 6 and Figure 7 show the output values from node O_2, O_3, O_4 and O_5.

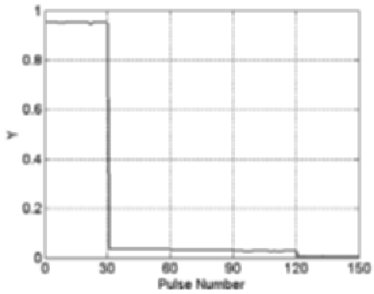


Figure 3: Simulated Output from Node O_1

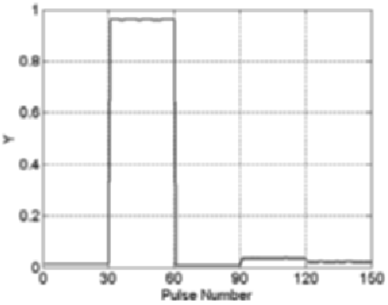


Figure 4: Simulated Output from Node O_2

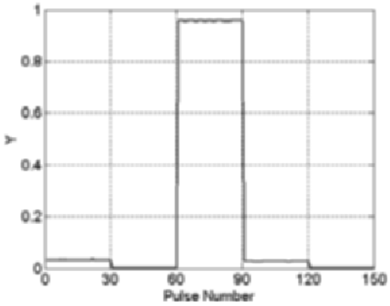


Figure 5: Simulated Output from Node O_3

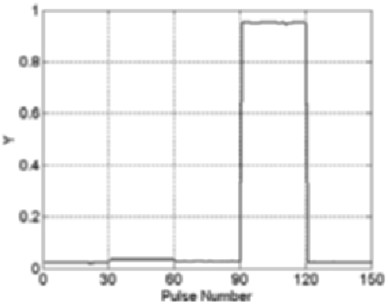


Figure 6: Simulated Output from Node O_4

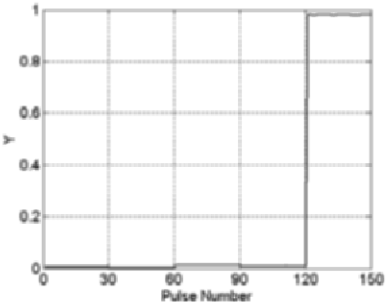


Figure 7: Simulated Output from Node O_5

7 Classification Accuracy

The CA of the each type was calculated using equation (1). Table 1 shows the classification accuracies for the five different pulse types mentioned above. The overall classification accuracy of the proposed neural network is 91%.

Table 1: Classification accuracies

Pulse Type	Classification Accuracy (%)
EC P	90.24
ECDP	88.05
SP	92.93
AP	89.31
SCP	95.43
Average	91.19

8 Process Control System

It can be observed that the various machining performances can be obtained by maintaining the proper percentages of the above mentioned pulse types. ECDPs and SPs are more favourable for fast metal removal rates whereas ECPs are more favourable for higher surface finish. However, APs and SCPs must be avoided, since they damage the work surface. Percentages of each type of pulses can be used to estimate the gap condition. The estimated gap condition can then be used as a feedback signal in the process control system. An algorithm must be developed to estimate the gap condition from the percentages of the pulse types.

9 Conclusions

In this paper, an ANN model for pulse classification in the ECDM process has been established and analysed based on the ECDM process variables. Designing stages such as feature extraction, preparation of training and test data set, selection of number of layers and number of neurons in the neural network and selection of activation function were described. Four features such as PV, AV, PC and AC were used successfully for the classification of pulses in the ECDM process. Classification accuracy has been defined to measure the accuracy of a pulse classification system. Simulation results showed that the feed forward network with six neurons in the hidden layer could be successfully used in pulse classification of the ECDM process.

References

Bermak, A. and Bouzerdoum, A. (2002), VLSI Implementation of a Neural Network Classifier Based on the Saturating Linear Activation Function, 9th International Conference on Neural Information Processing.

De Silva A.K.M, (1988), Process Developments in Electrochemical Arc Machining, PhD Thesis, University of Edinburgh.

De Silva A.K.M, Khayry A.B. and McGeough J.A. (1995), Process Monitoring and Control of Electroerosion-dissolution Machining, IMechE Conference Transactions, 11th International Conference on Computer-Aided Production Engineering, pp. 73-78.

Kao, J.Y. and Tarng, Y.S. (1997), A neural network approach for the on-line monitoring of the electrical discharge machining process, Journal of Material Processing Technology 69, pp. 112-119.

Liu, H.S. and Tarng, Y.S. (1997), Monitoring of the electrical discharge machining process by adaptive networks. International Journal of Advanced Manufacturing Technology 13, pp. 264-270.

Mediliyegedara, T.K.K.R., De Silva, A.K.M., Harrison, D.K., McGeough, J.A. (2004), An Intelligent Pulse Classification System for Electro Chemical Discharge Machining (ECDM) – A Preliminary Study, 14th International Symposium for Electromachining (ISEM-XIV), Edinburgh, U.K.

Pajak, E. and Wieczorowski, K. (1998), Classification of Discharges in Electocontact Discharge Machining (ECDM) by Means of Neural Networks, 12th International Symposium for Electro Machining (ISEM XII), Aachen, Germany, pp 225-232.

Tasi, K.M. and Wang, P.J. (2001), Comparisons of neural network models on material removal rate in electrical discharge machining. Journal of Materials Processing Technology, pp. 111-124.

Part IX

Signal Processing

Hybrid Image Segmentation based on Fuzzy Clustering Algorithm for Satellite Imagery Searching and Retrieval

W.S. Ooi and C.P. Lim*

School of Electrical & Electronic Engineering, University of Science Malaysia
Engineering Campus, 14300, Nibong Tebal, Penang, MALAYSIA
(*Email: cplim@eng.usm.my)

Abstract. Satellite image processing is a complex task that has received considerable attention from many researchers. In this paper, an interactive image query system for satellite imagery searching and retrieval is proposed. Like most image retrieval systems, extraction of image features is the most important step that has a great impact on the retrieval performance. Thus, a new technique that fuses color and texture features for segmentation is introduced. Applicability of the proposed technique is assessed using a database containing multispectral satellite imagery. The experiments demonstrate that the proposed segmentation technique is able to improve quality of the segmentation results as well as the retrieval performance.

Keywords: Image segmentation, hybrid color and texture feature, multispectral satellite imagery, fuzzy c-means clustering.

1 Introduction

Satellite imagery has been widely used in a variety of applications, e.g., classifying land uses, monitoring crop and forest harvests, tracking beach erosion, and determining regional geological structures. With advances in technology and the commercialization of imaging satellite, more and more countries are taking part in launching new satellites. As a result, the volume of satellite images continues to grow over the years. Thus an effective and efficient way of storing, managing, and retrieving a large collection of satellite images is increasingly becoming important.

Satellite images usually have a higher resolution than general-purpose images, making it extra difficult to index. In many applications, a complete and detailed analysis of the image content is often unnecessary.

Thus a way of extracting interested regions and index them in the database is needed. Segmentation is a useful method that is used to extract information from images. However, the presence of texture information in satellite images complicates the segmentation process. Classical segmentation techniques such as edge-based (Hueckel 1973) and histogram-based segmentation (Otsu 1978) algorithms often fail to partition a complex image into meaningful regions. To solve this problem, a hybrid method was presented to combine edge-based and region-based segmentation in aerial images segmentation (Nevatia and Price 1982). The segmentation processes are carried out independently and the results are then combined for further analysis and interpretation. Nevertheless, the approach incurs additional complexity and computational load. Recently, techniques that combine image features, e.g., color and texture for segmentation have received much attention from researchers (Chang and Wang 1996, Stefania et al. 1999, Ma and Manjunath 2000, Carson et al. 2002). These techniques have been extended to satellite application (Jolly and Gupta 1996) and the results are very promising. However, the problem of selecting the right features and the indexing method is still an open issue. More robust approaches to good segmentation are needed.

In this paper, a new approach that fuses color and texture features for image segmentation using the fuzzy c-means clustering algorithm (Bezdek 1981) is proposed. The proposed approach begins with the pre-processing of the input image, followed by the extraction of image features for segmentation, post-processed of the segmented result, and finally region description. The contribution of this paper is the systematic analysis and selection of features and the extraction techniques based on exhaustive empirical analysis. The resultant method has demonstrated improvement in quality of the segmentation results.

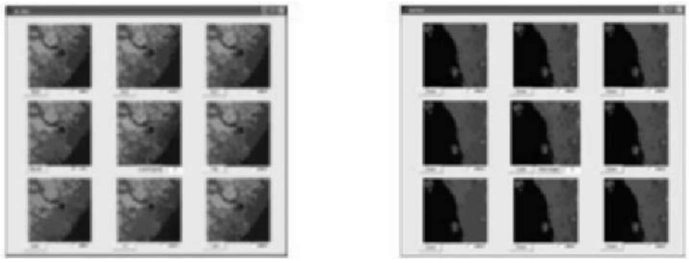
2 Image Segmentation and Feature Extraction Techniques

2.1 Color Feature Extraction

The effectiveness of color feature is dependent not only on the algorithm but also on the color space chosen. Thus far only a limited number of studies are conducted to compare the segmentation results using different color spaces (Zarit et al. 1999, Guo and Michael 2000). In this study, we have conducted a systematic experiment to determine the right color model that best fits our application. We explored most of the commonly used color spaces, i.e., the RGB, XYZ, YIQ, YCbCr, $I_1I_2I_3$, HSV, HIS, and CIELAB spaces. Color quantization has been omitted to reflect the detailed information on the segmented regions.

The performance of image segmentation can be quantified by measuring its consistency with human segmentations. In general, an ideally segmented image (standard model) is created by manual segmentation. However, creating standard model is a tedious (Zarit et al. 1999) and subjective task which is impossible for satellite images which have high resolution and complex scenes. Therefore, human judgment of the segmentation results is better than evaluation of the probability of error based on the image model.

Owing to the above reason, we propose a qualitative measurement method based on human subjective evaluation. In this method, 20 test images from two different collections, i.e., Landsat MSS (Landsat MSS Imagery 1998) and TM satellite imageries (Landsat TM Imagery 2003) are randomly selected for segmentation. The results are presented to users who have experience in image processing and graphic design. To avoid the human fatigue problem, each user was asked to rate only the top 3 images as compared to their original images. The color space that consistently led to high ratings was chosen. To make the experiment reliable, users were not explicitly told what color models were used to produce the results. After evaluating each test image, the subsequent test images were randomly allocated to avoid biased selection. Figure 1 shows the user interfaces used in our experiment. The center image is the original image and moving from top left in the clockwise direction are the segmentation results based on, respectively, the RGB, XYZ, $I_1I_2I_3$, YIQ, YCbCr, HIS, HSV and CIELAB color spaces. Figure 2 shows the evaluation results of two collections of test images, and the overall results are summarized in Fig 3.



(a) (b)

Fig. 1. The subjective test user interfaces for the evaluation of the segmentation results in different color spaces for (a) The Landsat MSS satellite image and (b) The Landsat TM satellite image.

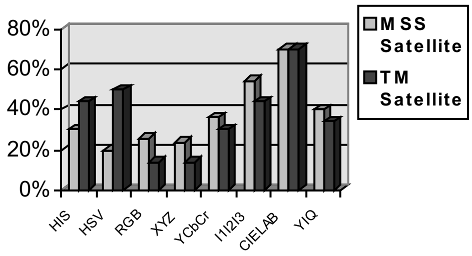


Fig. 2. The evaluation results in different color spaces for the Landsat MSS and TM satellite images

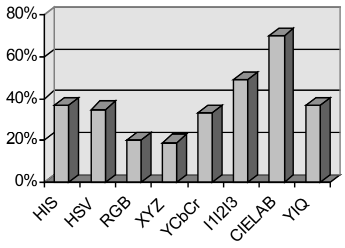


Fig. 3. The overall evaluation results in different color spaces

From the results in Fig. 3, it can be seen that 70% of the images of the CIELAB space were rated at the top 3 position. This was followed by the $I_1I_2I_3$ (49%), YIQ (37%), and HIS (37%) color spaces. The robustness of the CIELAB space might be owing to its perceptually uniform nature (Axiphos 2001). Hence, the CIELAB space was selected for further experimentation in our work.

To date, only a limited number of studies investigating the usefulness of luminance (*L*) and chromatic (*a* and *b*) components in image segmentation are available. Researches have shown that removal of the *L* component can tolerate shading artifacts and increases processing speed owing to dimensionality reduction (Liew et al. 1999, Liapis et al. 2000). However, Carson et al. (2002) and Dong et al. (2001) included the *L* component in image segmentation, but no justifications were provided. As use of the *L* component is dependent on applications, we therefore performed a test on the same test images to determine the usefulness of L component. The results, however, revealed that the *L* component should not be omitted but it could be coarsely quantized to reduce the shading artifacts. Several examples of poor segmentation results without the *L* component are illustrated in Fig. 4.

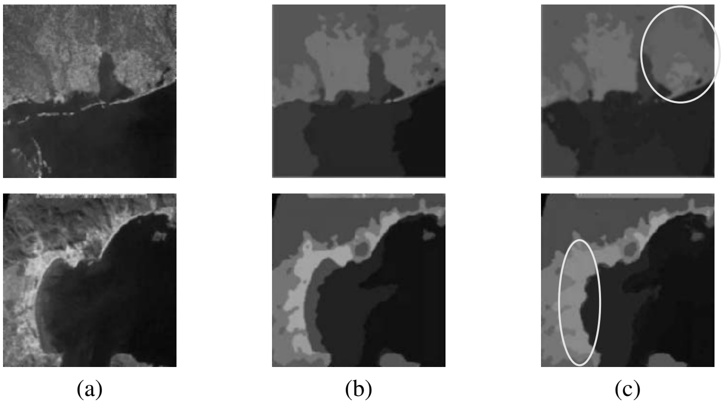


Fig. 4. Image segmentation (a) Original image (b) Luminance and chromatic components are used (c) only chromatic component is used. The yellow circles show the false segmentation obtained when only chromatic component is used.

2.2 Texture Feature Extraction

Texture is another important property for the analysis of satellite imagery. The Grey-Level Co-occurrence Matrix (GLCM) (Haralick 1979) is one of the most popular statistical approaches used in texture discrimination. Several researchers (Ohanian and Dubes 1992, Sharma and Singh 2001) have reported that the co-occurrence statistics outperformed other texture extraction techniques. One of the criticisms of the GLCM is its high computational demand. As a result, we have employed the Grey Level Co-occurrence Linked List (GLCLL) (Clausi and Jernigan 1998) approach. In this approach, the texture window is shifted in a zigzag manner and only the changes of non-zero probabilities of grey level pairs are updated.

The problem of selecting the appropriate texture descriptors is still an open issue. A set of fourteen texture descriptors was proposed to characterize the co-occurrence matrix contents (Haralick 1979). However, not all descriptors are used in practice. We therefore performed a series of empirical tests to determine the descriptors that best fit our application. We employed the most commonly used descriptors suggested by Barber and LeDrew (1991) and Hall-Beyer (2000) for analysis i.e., contrast, dissimilarity, homogeneity, energy, entropy, and correlation. We conducted three experiments to compare the segmentation results. The same set of test images used in the previous experiment was employed. First, we compared the results for each texture descriptor fused with the CIELAB color components. The second test combined a pair of descriptors with the CIELAB color components. The combination yielded

15 results for each image under test. In the final test, we extended the combination to three texture descriptors combined with the CIELAB color components to produce 20 results for each test image. Owing to limited space, only the results of the first test are illustrated in Fig. 5.

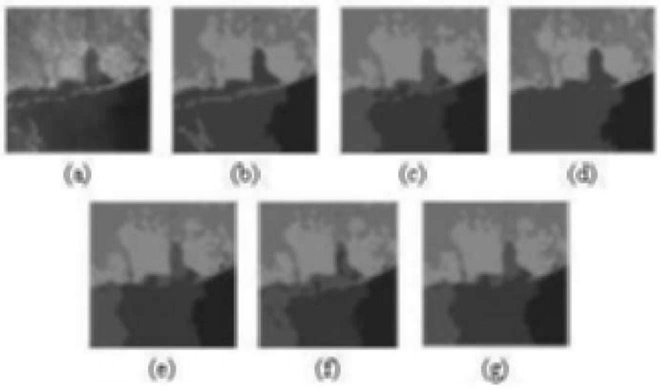


Fig. 5. Image segmentation results (a) Original image. The results when the color components are combined with each texture descriptors (b) Contrast (c) Entropy (d) Dissimilarity (e) Homogeneity (f) Energy and (g) Correlation.

From the experiment, we found that pairwise combination of texture descriptors yielded good results. Adding more descriptors, however, did not improve the results significantly but would induce a higher computational load. Among different descriptors used, the contrast measure gave good discrimination power, as shown in Fig. 5. However, the best result was obtained when contrast and entropy measure were combined for segmentation. The incorporation of the entropy measure can overcome the missing information detected by contrast statistic alone.

Co-occurrence matrices can be constructed easily in a grayscale or monochrome image. However, it must be generalized for the application of multispectral satellite images. A color image can be modeled as three-band monochrome image data. At first sight, it seems reasonable to construct co-occurrence matrices from each color band. However, a number of issues that worth studying include which color band is the most suitable one for extraction? Does the combined textural information from three different color bands provide optimal results? Is there any correlation between color bands that needs to be considered when texture information is being captured? Very few investigations have been performed to address these issues. In Palm et al. (2000), two methods were proposed to extend the grayscale co-occurrence statistics to color image. The first method, known as within co-occurrence matrices interprets each color band as a grayscale image. The computed co-occurrence texture features from each color band are combined for analysis. The second method computes the cross co-occurrence matrices that consider the pixel relationship of different color bands. Due to the transition between different bands during computation, the matrices lose their symmetry. The proposed methods outperformed the conventional GLCM approach. The results further showed that the within co-occurrence matrices outperformed the cross co-occurrence matrices.

In this study, we extended the comparison by incorporating more combination of color bands. Unlike previous works, the formulation of texture features was performed on the CIELAB color space. By using the same set of test images, we extracted the texture and color information in the following ways:

1. *L* for texture, *a* and *b* for color
2. *L* for texture, *L*, *a*, and *b* for color
3. Grayscale for texture, *L*, *a*, and *b* for color
4. *L*, *a*, and *b* for texture and color (the within co-occurrence matrices extended to CIELAB space)

Prior to texture extraction, the *L* component and the grayscale version were quantized into 16 levels to reduce computational time and to improve the statistical validity (Hall-Beyer 2000).

Using the above combination schemes, 80 segmentation results were generated. Several results are depicted in Fig. 6. We conducted the evaluation by manually marking the false segmented regions detected on each result. The best combination should accumulate the minimum marking. It can be observed that the first combination scheme produced the most misclassified regions. As mentioned earlier, false segmentations are likely to happen when only the chromaticity component is employed. Although texture information is incorporated, it does not help much in this case. The third combination scheme requires the transformation of RGB image to the Y component of the YUV color model. The Y component is a good grayscale representation of the color image. Surprisingly, this method performed slightly worse than the previous scheme where the *L* component was used. The segmentation results based on the second and fourth combination schemes are very similar. Both combination schemes are able to discriminate objects accurately. Because of this observation, the second combination scheme which has a lower computational overhead has been selected for further studies.

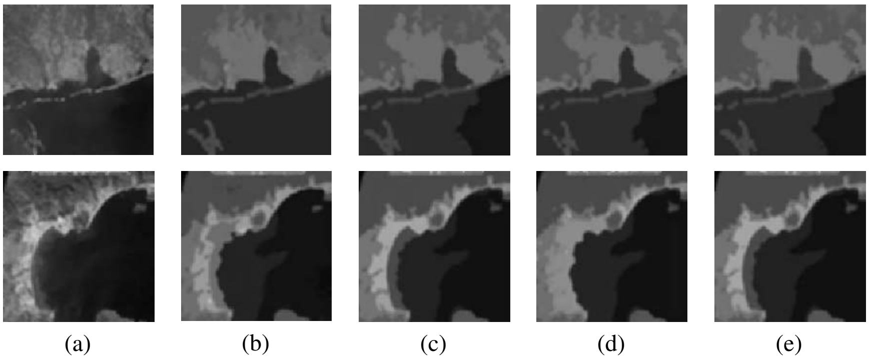


Fig. 6. Image segmentation results (a) Original images. The results from (b) to (e) are based on the above mentioned combination schemes.

2.3 Fuzzy C-means Clustering

Using the above feature extraction schemes, color and texture features from an image can be successfully extracted. As a result, a pixel in an image can be represented by three colors and two texture values. These features are combined to form a set of five-dimensional feature vectors. The next step will be the application of a suitable clustering technique to cluster individual pixels into groups that exhibit homogeneous properties.

The main difference between color and texture features is that the latter exhibits a local neighborhood property rather than point property. The computation of the texture feature of a pixel is commonly done in a $M \times N$ window. As a result the adjacent regions may not have sharp transitions of boundaries. If this is the case, fuzzy clustering is a solution, as pointed out by Krishnapuram (1998). In this study, fuzzy c-means (FCM) clustering is employed. It is an iterative algorithm that produces optimal partitions based on the minimization of the following objective function,

$$J_m = \sum_{i=1}^{c} \sum_{j=1}^{n} (\mu_{ij})^m \left\| X_j - V_i \right\|^2 \tag{1}$$

where $X_j \left(j = 1,\ 2...\ n\right)$ is the data point; V_i is the fuzzy centroid; μ_{ij} is the degree of membership of X_j in the cluster i; c is the number of clusters that needs to be determined in advance; n is the number of data points to be clustered; and m is the fuzziness index, which is set to a value greater than one. The norm operator $\|\cdot\|$ represents the standard Euclidean distance although can be generalized to other cases.

The CIELAB space is a perceptually uniform space. A perceptual difference between two colors is the distance between the color locations. This distance can be computed by the Euclidean distance measure. However when the Euclidean distance is employed, setting the tolerance limits of each component to its extreme limit, e.g. L* = 0 - 100, a* = b* = −120 - 120, resulting a box-shaped tolerance whereby the actual tolerances perceived by human resembles an ellipsoid (Axiphos 2001). To alleviate this problem, several equations were proposed, e.g., the CIELCH, CMC, and CIE94 color difference equations (Axiphos 2001). Continuing the revision of older equations, CIEDE2000 was proposed (Luo et al. 2001) to improve the classical equations. We have therefore modified the distance function to suit our algorithm as follows

$$J_m = \sum_{i=1}^{c} \sum_{j=1}^{n} (\mu_{ij})^m d(X_j, V_i) \tag{2}$$

with

$$d(X_j, V_i) = w_1 \left\| X_j^{con,ent} - V_i^{con,ent} \right\| + w_2 \cdot CIEDE2000\left(X_j^{Lab} - V_i^{Lab}\right)$$

where *con* is contrast; *ent* is entropy; w_1 and w_2 are the feature weights which define how significant the effect of texture or color in segmentation. In our case w_1=0.8 and w_2=1 give optimal results. To determine the optimal number of clusters for the clustering algorithm, we employed the iteration method presented in (Xuanli and Gerardo 1991).

2.4 Region Merging and Labeling

After pixels grouping by FCM, the resulting output matrix is a set of image regions labeled by region number (1, 2,..., c). As an example, suppose we have the output matrix as shown in Fig. 7(a), these image regions may consist of several disconnected sub-regions. The next task is to identify which sub-regions should be connected based on the connectivity criteria. Any two pixels (x_a, y_a), (x_b, y_b) are considered to be connected if they share the similar intensity value V and satisfy the adjacency criterion for connectivity. For a binary image $V = \{1\}$, for grayscale image V might be any range of grey levels, i.e., $0 \le V \le 255$. For a pixel with the coordinate (x, y), its 4-connectivity is the set that includes its horizontal and vertical neighbors:

$$N_4((x,y)) = \{(x+1,y),(x-1,y),(x,y+1),(x,y-1)\} \tag{3}$$

The 8-connectivity of (x,y) is a superset of the 4-connectivity and contains the horizontal, vertical and diagonal neighbors:

$$N_8((x,y)) = N_4((x,y)) \cup \begin{Bmatrix} (x+1,y+1),(x-1,y-1), \\ (x+1,y-1),(x-1,y+1) \end{Bmatrix} \tag{4}$$

For the following discussion, we use 8-connectivity. The labeling algorithms scan every pixel in the each row and label connected regions with the same number. The result after grouping and labeling sub-regions is shown in Fig. 7(b). Note that those regions labeled 1 are re-labeled to show discontinuity.

The next step will be merging of over segmented regions. Image smoothing is always performed a priori to avoid over-segmentation. However, image smoothing should be omitted during texture extraction since the intensity variation of dissimilar texture regions will be averaged to become a homogeneous region by smoothing, which will lead to under-segmentation. Thus implementing region merging is one of the best ways to overcome this problem. The merging of two adjacent regions, R_a and R_b is performed based on the uniformity criterion, i.e., the distance measure of the mean color $\mu_c^{a,b}$ and the mean texture $\mu_t^{a,b}$ values between these regions is below a pre-specified threshold, T_m. The new region formed take the average properties of R_a and R_b. Referring to the example, assume that R_1 and R_2 satisfy the merging criterion, the resulting matrix after merging is shown in Fig. 7(c).

After merging the over-segmented regions, the next step is to reduce the non-dominant region. Truncation of small regions can help to increase the retrieval speed and accuracy. Truncation is performed when the $i-th$ region size is less than a threshold T_a,

$$A_i < T_a \tag{5}$$

where A_i denotes the area of region R_i. The threshold T_a is the ratio of A_i over the largest region area in the image. Assume that T_a is 10% of the largest region area, then region R_4 satisfies the truncation criteria.

Assume further that $\frac{1}{2}\cdot\left(\mu_c^{3,4}+\mu_t^{3,4}\right)<\frac{1}{2}\cdot\left(\mu_c^{1,4}+\mu_t^{1,4}\right)<\frac{1}{2}\cdot\left(\mu_c^{5,4}+\mu_t^{5,4}\right)$, R_4 will merge to R_3 and the resulting image is shown in Fig. 7(d). At the end of this operation, the image regions are re-labeled orderly for subsequent features extraction.

1	2	1	3	1	1
1	2	1	3	3	1
1	1	2	2	4	3
1	2	2	2	2	4
1	1	2	2	2	2

(a)

1	2	1	3	5	5
1	2	1	3	3	5
1	1	2	2	4	3
1	2	2	2	2	4
1	1	2	2	2	2

(b)

1	1	1	3	5	5
1	1	1	3	3	5
1	1	1	1	4	3
1	1	1	1	1	4
1	1	1	1	1	1

(c)

1	1	1	3	5	5
1	1	1	3	3	5
1	1	1	1	3	3
1	1	1	1	1	3
1	1	1	1	1	1

(d)

Fig. 7. Region merging and labeling example (a) Output matrix after FCM (b) Output matrix after 8-connected component. (c) After region merging. (d) After trimming of small size region.

2.4 Region Feature Extraction

With regards to satellite image browsing, color, texture, and shape are the primary features used for region discrimination. As color is the most important clue in multispectral images, we stored the top two dominant colors of each region. The texture values of every pixel in the region are average and indexed. To capture the geometrical properties of segmented object, we computed the area, center of area (centroid), and eccentricity features. These features are simple and fast to index.

Computing the area of an image region is particular useful in satellite application since this property can reflect the actual size of a region on the Earth's surface. We can define the area of the $i-th$ object as

$$A_i=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}I_i(x,y) \tag{6}$$

where

$$I_i(x,y)=\begin{Bmatrix}1 & if\ I(x,y)=ith\ object\ number\\ 0 & otherwise\end{Bmatrix}$$

The center of area (centroid) specifying the location of an object and can be defined as

$$Centroid=\left(\overline{x_i},\overline{y_i}\right) \tag{7}$$

where

$$\overline{x_i} = \frac{1}{A_i}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} x \cdot I_i(x,y) \text{ and } \overline{y_i} = \frac{1}{A_i}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} y \cdot I_i(x,y)$$

There are several measures of eccentricity. We employed the most common used measures which can be defined by the ratio of the length of maximum chord A to maximum chord B perpendicular to A , as illustrated in Fig. 8.

$$E_i = \frac{A_i}{B_i} \tag{8}$$

Having undergone the feature extraction phase, our feature vector contains twelve features measures, which is sufficient to represent an object in the satellite application.

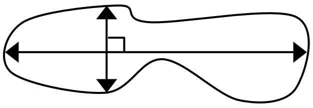


Fig. 8. Eccentricity

3 Experimentation

Two experiments were conducted to evaluate the performance of the proposed image segmentation algorithm. The first experiment demonstrates the usefulness of integrating color and texture information in image segmentation. The same image in Fig. 5(a) was applied for demonstration. The optimal number of clusters computed for this image is c =5. We performed segmentation on this image based on color, texture and their combination, and the results were compared with the original image.

The goal of the second experiment is to evaluate the effectiveness of the proposed method applied to satellite imagery search and retrieval. The database used contained 400 satellite images. The images are classified into two main categories. The first category of images was taken from the four bands of the Multispectral Scanner (MSS) sensors carried by older Landsat 1, 2 and 3 (Landsat MSS Imagery 1998). The second category of images is the latest Landsat 7 Thematic Mapper (TM) images. The Landsat TM images were obtained from (Landsat TM Imagery 2003). Image segmentation and feature extraction of all the images were performed off-line. After segmentation, more than 4000 regions were obtained.

The querying step is straightforward. Given a query image, the same segmentation operation is applied to obtain the segmented regions. By selecting the region to match, the user can specify the weights for each region feature. Specifying feature weights can lead to faster convergence towards the target images. Moreover, user is able to understand the retrieval process and the returned results better as compared to automated weight adjustment approach. Once the query is submitted, the system retrieves the images from the image database which are similar to the query image. The similarity between two regions is defined by

$$d(overall) = \frac{1}{4}\begin{bmatrix} w_c \cdot d_n(color) + w_t \cdot d_n(texture) + \\ w_s \cdot d_n(shape) + w_l \cdot d_n(location) \end{bmatrix} \tag{9}$$

where

$$d_n(color) = \Delta E^*_{2000}\left(L^*, a^*, b^*\right)$$

$$d_n(texture) = \left[(\Delta contrast)^2 + (\Delta entropy)^2\right]^{\frac{1}{2}}$$

$$d_n(shape) = \left[(\Delta area)^2 + (\Delta elongation)^2\right]^{\frac{1}{2}}$$

$$d_n(location) = \left[(\Delta x)^2 + (\Delta y)^2\right]^{\frac{1}{2}}$$

The color distance is computed using the CIEDE2000 equation (Luo et al. 2001). The subscript n denotes the normalization of vector component between 0 and 1. This prevents measurement bias when different vector components with varying range are used. The weight, w, is set by the user during the query composition. The default weights are set as $w_c = w_t = w_s = w_l = 1$.

The proposed system was compared with two classical methods. The first method was the global histogram matching introduced by Swain and Ballard (1991). The color histogram for each image was computed based on the quantized (166 bins) RGB color space. The second method divided an image into 4×4 sub-images. The color histogram for each sub-image was computed and indexed for subsequent matching. The second classical method has been widely used in image retrieval (Schettini et al. 2001). However, the computation load increased considerably due to the repetition indexing of divided sub-images. The comparison algorithms were coded using the MATLAB programming language (Rudra 2001). An interactive interface that incorporates the image query system has been developed, as shown in Fig. 9.

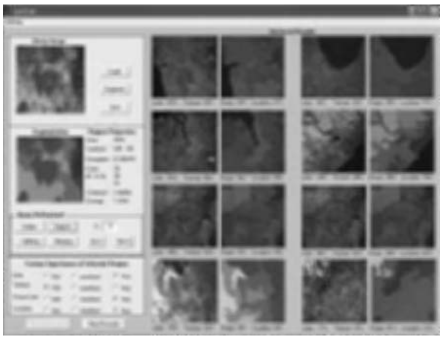
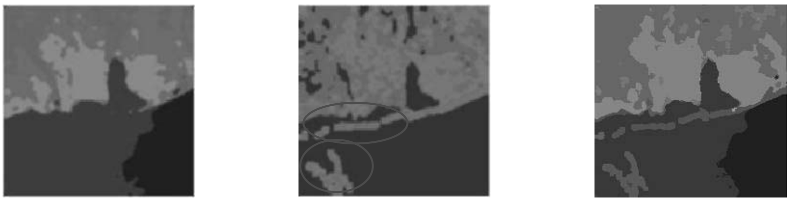


Fig. 9. The image retrieval system user interface. The representation of the query image (bottom of query image) and the returned images (left of the retrieved images) are produced to help the user understand how the system reasons.

4 Results and Discussion

The results of the first experiment are presented in Fig. 10. As shown in Fig. 10(a), smoother segmented regions can be obtained but missing data are likely to occur when the color feature alone is used for segmentation. In contrast, the segmented regions based on the texture feature alone are scatter but regions (red circles) undetectable by using color segmentation can be identified as illustrated in Fig. 10(b). The fusion of color and texture features produces satisfactory results. As illustrated in Fig. 10(c), after region merging and labeling, blue regions have been successfully segmented.



(a) (b) (c)

Fig. 10. Image segmentation results based on (a) color feature alone (b) texture feature alone (c) the fusion of color and texture features in segmentation improve the segmentation results.

To measure the effectiveness of the retrieval systems, recall and precision measurements were employed (Jones 1981). For each category of images we tested the queries for vegetation and urban regions. All the images were manually inspected for these two queries. In general, 70% of the database images contain vegetation. However, the vegetation regions in some images are relatively small. Thus, it becomes difficult to judge whether the retrieved vegetation regions are relevant to the query in the histogram-based approach. To make the test reliable, we defined the retrieved regions as relevant when the relative size fell below 50% tolerance limits of the query region size. We did not set any constraint on the urban query since urban regions are relatively small in our dataset. We performed five browsing sessions on each query with the results averaged.

As shown in Fig. 11, the proposed method outperformed the two classical methods in almost all of tests. The results indicate that the classical methods performed rather well in the vegetation queries. This might be owing to vegetation areas occupied a large portion of the relevant images. As a result, the histogram-based methods were able to sustain a high precision rate even with the lack of ability to capture spatial information. However, our method was significantly better for queries on urban areas. Unlike vegetation areas, the urban areas were relatively small and sometimes randomly distributed. In this case, global histogram yielded the worst precision rate. As we expected, the local histogram method performed better than the global histogram method due to its ability to localize image objects.

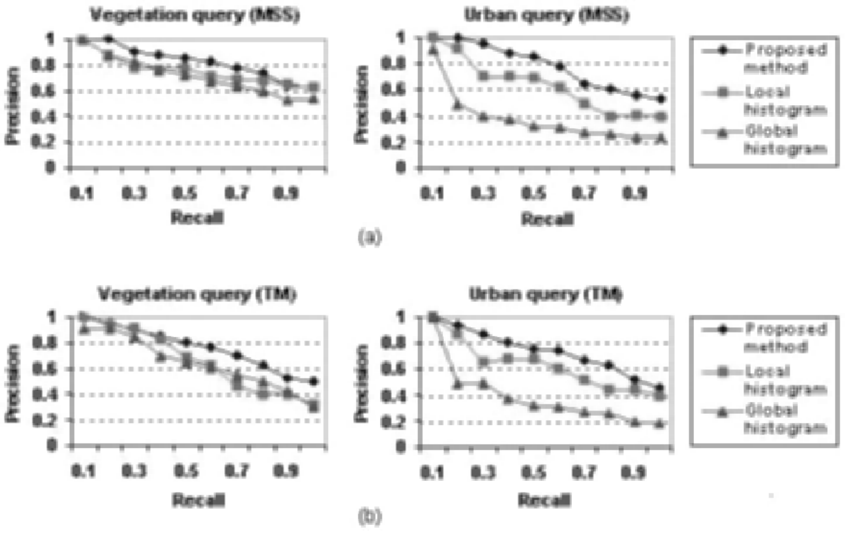


Fig. 11. The recall vs. precision graph of vegetation and urban queries for (a) Landsat MSS satellite imagery (b) Landsat TM satellite imagery.

5 Summary

A new approach to image segmentation in satellite imagery application based on color and texture fusion is presented. The segmentation process consists of three stages of computation: extraction of color and texture features for each image pixel, grouping pixels into regions that exhibit homogeneous properties, and describing visual properties of each region for use in a query. We conducted a thorough experimental analysis to determine the suitable image features. A modification of the fuzzy c-means distance function used in pixel classification was proposed. We demonstrated that the proposed segmentation algorithm outperformed classical methods which treat color or texture features independently. We further justified the effectiveness of the approach applied to satellite imagery searching and retrieval. The experimental results ascertained that the proposed approach is effective, and it compares favorably with classical approaches.

Acknowledgments

The authors acknowledge the research grants provided by Universiti Sains Malaysia as well as the Ministry of Science, Technology, and Innovation Malaysia (No. 06-02-05-8002 & 04-02-05-0010) that have in part resulted in this article.

References

Axiphos GmbH, A Marketing, Trading and Consulting Company, GERMANY (2001) On color differences formulae. Technical Report

Barber DG, LeDrew EF (1991) SAR sea ice discrimination using texture statistics: a multivariate approach. Photogrammetric Engineering & Remote Sensing 57, no. 4: 385-95

Bezdek JC (1981) Pattern recognition with fuzzy objective function algorithms. NY: Plenum Press

Carson C, Belongie S, Greenspan H, Malik J (2002) Blobworld: image segmentation using expectation-maximization and its application to image querying. IEEE TPAMI, 24(8):1026-1038

Chang CC, Wang LL (1996) Color texture segmentation for clothing in a computer-aided fashion design system. Image and Vision Computing 14, no. 9, pp 685-702

Clausi DA, Jernigan ME (1998) A fast method to determine co-occurrence texture features. IEEE Transactions on Geoscience and Remote Sensing 36 (1), pp 298-300

Clausi DA, Zhao Y (2002) Rapid co-occurrence texture feature extraction using a hybrid data structure. Computers & Geosciences 28 (6), pp 763-774

Clausi DA, Zhao Y (2003) Grey level co-occurrence integrated algorithm (GLCIA): a superior computational method to determine co-occurrence texture features. Computers and Geosciences, vol. 29, no. 7, pp 837-850

Dong G, Boey KH, Yan CH (2001) Feature discrimination in large scale satellite image browsing and retrieval. 22nd Asian Conference on Remote Sensing. vol. 1, pp 203-207

Guo P, Michael RL (2000) A study on color space selection for determining image segmentation region number. Proc. of the 2000 International Conference on Artificial Intelligence (IC-AI'2000), Monte Carlo Resort, Las Vegas, Nevada, USA, vol. 3, pp 1127-1132

Hall-Beyer M (2000) GLCM texture: a tutorial. NCGIA remote sensing core curriculum. Retrieved January 14, 2001, from http://www.cla.sc.edu/geog/rslab/rsccnew/rscc-frames.html

Haralick RM (1979) Statistical and structural approaches to texture. Proc. of the IEEE, 67:786-804

Hueckel M (1973) A local visual operator which recognized edges and lines," Journal of the Association for Computing Machinery 20, pp 634-647

Jolly MPD, Gupta A (1996) Color and texture fusion: application to aerial image segmentation and GIS updating. Proc. Third IEEE Workshop on Applications of Computer Vision, pp 2-7

Jones KS (1981) Information retrieval experiment. Butterworth and Co

Krishnapuram R (1998) Segmentation. Section on "Computer Vision" in Handbook of Fuzzy Computation, E. Ruspini, P. Bonissone, and W. Pedrycz (Ed.), Institute of Physics Publishing, pp F7.4.1-F7.4.5

Landsat MSS Imagery: About “LANDSAT Images of the U.S.A Archive” (1998). Retrieved April 13, 2002, from http://www.nasm.si.edu/ceps/rpif/landsat/Viewing.html

Landsat TM Imagery: Malaysian Centre for Remote Sensing (MACRES) (2003). Retrieved June 11, 2003, from http://www.macres.gov.my

Liapis S, Sifakis E, Tziritas G (2000). Color and/or texture segmentation using deterministic relaxation and fast marching algorithms. Intern. Conf. on Pattern Recognition, vol. 3, pp 621-624

Liew WC, Sum KL, Leung SH, Lau WH (1999) Fuzzy segmentation of lip image using cluster analysis. Proc. of Eurospeech'99, vol. 1, pp 335-338

Luo MR, Cui G, Rigg B (2001). The development of the CIE 2000 colour difference formula: CIEDE2000. Color Res. Appl., 26, pp 340-350

Ma WY, Manjunath BS (2000) EdgeFlow: a technique for boundary detection and image segmentation. IEEE Trans. Image Processing, 9(8):1375-1388

Nevatia R, Price KE (1982) Locating structures in aerial images. IEEE Transactions on PAMI, Volume PAMI-4, Number 5, pp 476-484

Ohanian PP, Dubes RC (1992) Performance evaluation for four class of texture features. Pattern Recognition, vol. 25, no. 8, pp 819-833

Otsu N (1978) A threshold selection method from grey-level histograms. IEEE Trans. Syst., Man, Cybern., vol. SMC-8, pp 62-66

Palm C, Lehmann T, Spitzer K (2000) Color texture analysis of moving vocal cords using approaches from statistics and signal theory. In: Braunschweig T, Hanson J, Schelhorn-Neise P, Witte H: Proceedings of the 4th International Workshop: Advances in Quantitative Laryngoscopy, Voice and Speech Research, Friedrich-Schiller University. Jena, pp 49-56

Rudra P (2001), Getting started with Matlab: Version 6: a quick introduction for scientists and engineers. Oxford University Press

Schettini R, Ciocca G, Zuffi S (2001) A survey on methods for colour image indexing and retrieval in image databases. Color Imaging Science: Exploiting Digital Media, (R. Luo, L. MacDonald eds.), J. Wiley

Sharma M, Singh S (2001) Evaluation of texture methods for image analysis. Proc. 7th Australian and New Zealand Intelligent Information Systems Conference, Perth, pp 117-121

Stefania A, Ilaria B, Marco P (1999) Windsurf: region-based image retrieval using wavelets. DEXA Workshop, pp 167-173

Swain MJ, Ballard DH (1991) Color indexing. IJCV, vol. 7, no. 1, pp 11-32

Xuanli LX, Gerardo B (1991) A validity measure for fuzzy clustering. TPAMI, 13(8):841-847

Zarit BD, Super BJ, Quek FKH (1999) Comparison of five color models in skin pixel classification. Proc. Intl. Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, pp 58-63

Boosting the Performance of the Fuzzy Min-Max Neural Network in Pattern Classification Tasks

Kok Yeng Chen[1], Chee Peng Lim[1]*, and Robert F. Harrison[2]

School of Electrical and Electronic Engineering[1]
University of Science Malaysia, Engineering Campus
14300 Nibong Tebal, Penang, Malaysia
Email: (corresponding author*) cplim@eng.usm.my

Dept of Automatic Control & Systems Engineering[2]
The University of Sheffield
Mappin Street, Sheffield S1 3JD
United Kingdom

Abstract. In this paper, a boosted Fuzzy Min-Max Neural Network (FMM) is proposed. While FMM is a learning algorithm which is able to learn new classes and to refine existing classes incrementally, boosting is a general method for improving accuracy of any learning algorithm. In this work, AdaBoost is applied to improve the performance of FMM when its classification results deteriorate from a perfect score. Two benchmark databases are used to assess the applicability of boosted FMM, and the results are compared with those from other approaches. In addition, a medical diagnosis task is employed to assess the effectiveness of boosted FMM in a real application. All the experimental results consistently demonstrate that the performance of FMM can be considerably improved when boosting is deployed.

Keywords: Pattern Classification, Fuzzy Min-Max Neural Network, AdaBoost

1. Introduction

Pattern classification is an important area that has great application to various real world problems, e.g. diagnostic applications such as medical diagnosis and machine fault diagnosis. Many aspects of pattern classification have been studied extensively and some good references include Devijver and Kittler (1982), Duda and Hart (1973) and Fukunaga (1972).

In recent years, artificial neural networks have shown their abilities in pattern classification tasks, e.g. speech recognition, classifications of handwritten characters, and fault detection and diagnosis. Among various neural network models, the Fuzzy Min-Max neural network (FMM) (Simpson 1992) is an example of a learning algorithm which is able to learn in a single pass through the data samples and, at the same time, to build and fine-tune the decision boundaries of different classes online. In order to improve the performance of a learning algorithm, ensemble methods have often been used. A promising method is boosting which attempts to improve the accuracy of a weak learning algorithm by repeating the training procedure many times and by taking the "majority vote" of the output hypotheses. Boosting modifies the distribution of samples in order to force the weak learning algorithm to concentrate on the "hard" examples. A number of research results have shown the various characteristics of boosting in improving the performance of a single classifier, e.g. Breiman (1996) and Freund and Schapire (1996).

In this paper, we investigate an approach that combines AdaBoost (Freund & Schapire, 1997), a favorable boosting algorithm, and FMM and apply the resulting boosted FMM to two benchmark databases as well as a real medical diagnosis problem. The organization of this paper is as follows. Section 2 gives an introduction to FMM, as well as boosting with specific attention to the AdaBoost algorithm. The boosted FMM framework is also presented. Section 3 presents the experimental results and discussion. Finally, conclusions and suggestions for future work are presented in Section 4.

2. Fuzzy Min-Max Neural Network and AdaBoost

2.1 Fuzzy Min-Max Neural Network

The FMM network has the ability to add new pattern classes online and to refine the existing pattern classes as new data samples are received. The main components of this neural network model are the hyperboxes that are created and defined by pairs of min-max points and their corresponding membership functions that describe the degree to which an input pattern fits within the min-max points. Figure 1 shows a three-layer FMM network while Figure 2 illustrates the segregation of several (hyper) rectangular boxes for a two-class problem in a two-dimensional space.

In general, each F_B node represents a hyperbox fuzzy set where the F_A to F_B connections are the min-max points and the F_B transfer function is the hyperbox membership function. Each hyperbox fuzzy set, B_j, is defined by

$$B_j = \{A, V_j, W_j, f(A, V_j, W_j)\} \qquad \forall A \in I^n \tag{1}$$

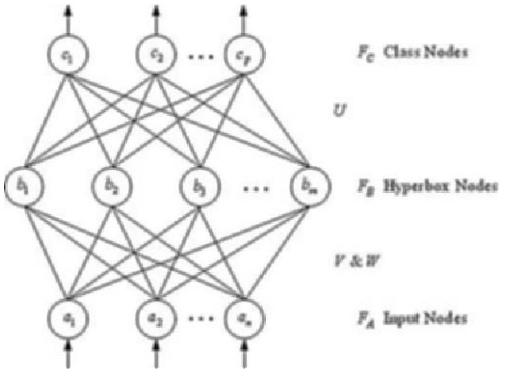


Fig. 1. The topology of the FMM

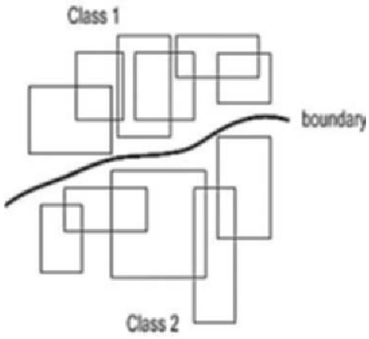


Fig. 2. Decision boundary established by the FMM for a two-class problem in a two-dimensional space

where $A = (a_1, a_2, \ldots\ldots, a_n)$ is the input pattern $V_j = (v_{j1}, v_{j2}, \ldots\ldots, v_{jn})$ and $W_j = (w_{j1}, w_{j2}, \ldots\ldots, w_{jn})$ are the minimum and maximum points of B_j. The membership function, $f(A, V_j, W_j)$, is defined by

$$b_j(A_h) = \frac{1}{2n}\sum_{i=1}^{n}[\max(0, 1-\max(0, \gamma\min(1, a_{hi} - w_{ji}))) + \max(0, 1-\max(0, \gamma\min(1, v_{ji} - a_{hi})))] \tag{2}$$

where A_h is the h-th input pattern, γ is the sensitivity parameter that regulates how fast the membership values decrease as the distance between A_h and B_j increases.

The membership function indicates the measurement of how far each component is greater (lesser) than the maximum (minimum) point value along each dimension. If $b_j(A_h) \to 1$, the point should be more "contained" by the hyperbox. When $b_j(A_h) = 1$, it represents complete hyperbox containment. Each F_C node represents a class, and output of the F_C node represents the degree to which the input pattern A_h fits within class *k*. The connections between the F_B

and F_C nodes are binary valued and are stored in matrix U as shown in equation (3)

$$u_{jk} = \begin{cases} 1 & \text{if } b_j \text{ is a hyperbox for class } c_k \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where b_j is the j-th F_B node and c_k is the k-th F_C node. The transfer function for each F_C node performs the fuzzy union operation of the appropriate hyperbox fuzzy set values as defined by

$$c_k = \max_{j=1}^{m} (b_j \times u_{jk}) \tag{4}$$

Thus the class whose hyperbox is best suited to the input vector is chosen as the predicted class for the input pattern.

The FMM learning algorithm comprises an expansion/contraction process. The algorithm first searches a hyperbox of the same class that can expand (if necessary) to include the incoming input pattern. If the hyperbox meets the expansion criteria, as described in equation (5), the input pattern will be included in the hyperbox. Otherwise, a new hyperbox is formed and added to the network. The maximum size of a hyperbox is bounded by $0 \le \theta \le 1$, a user defined value.

$$n\theta \ge \sum_{i=1}^{n} \left(\max(w_{ji}, a_{hi}) - \min(v_{ji}, a_{hi}) \right) \tag{5}$$

In summary, the FMM dynamics comprise a three-step process:

1. ***Expansion***: Identify the hyperbox that can expand and expand it. If an expandable hyperbox cannot be found, a new hyperbox for that class will be added.
2. ***Overlap Test***: Determine if any overlap exists between hyperboxes from different classes.
3. ***Contraction***: If overlap between hyperboxes of different classes does exist, eliminate the overlap by minimally adjusting each of the hyperboxes.

The detailed description of the FMM can be found in Simpson (1992).

2.2 AdaBoost

Boosting has its roots in the theoretical framework for studying machine learning called the "PAC" (*Probably Approximately Correct*) learning model (Valiant 1984). Kearns and Valiant (1988, 1994) were the first to pose the question of whether a "weak" learning algorithm which performs just slightly better than random guessing in the PAC model can be boosted into an arbitrarily accurate, strong learning algorithm. Schapire (1990) came up with the first provable polynomial-time boosting algorithm. Later, Freund and Schapire (1997) developed a more efficient boosting algorithm called AdaBoost that had solved many practical difficulties of the early boosting algorithms.

To achieve a higher accuracy rate than the weak learner's hypothesis, AdaBoost executes the weak or base learning algorithm in a series of rounds on slightly altered training data, and combines the hypotheses into one final decision. The main idea of AdaBoost is to assign a weight to each sample in the training set. At the early stage of the process, all weights are equal. But, in every round the weak learner returns a hypothesis, and the weights of all samples that are wrongly classified by the hypothesis are increased. The idea of this process is to force the weak learner to focus on the difficult samples in the training set. The final hypothesis is a combination of the hypotheses of all rounds, namely a weighted majority vote, where hypotheses with lower classification error would have higher weights. The procedure of AdaBoost is as follows, and its detailed description can be found in Freund and Schapire (1997, 1999).

```
Given: $(x_1, y_1)$, ……., $(x_m, y_m)$ where $x_i \in X,\ y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = \frac{1}{m}$, where $m$ = number of inputs
For $t$ = 1, . . .,$T$ :
    1. Train the weak learner using distribution, $D_t$.
    2. Get weak hypothesis, $h_t : X \to \{-1, +1\}$, and calculate the weighted
    error of $h_t$ :
            $\varepsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i]$
        Abort the loop if $\varepsilon_t > 0.5$
    3. Choose $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$
    4. Update distribution, $D_t$
```

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \text{ (correctly classified)} \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \text{ (incorrectly classified)} \end{cases}$$

$$= \frac{D_t(i)\exp\left(-\alpha_t y_i h_t(x_i)\right)}{Z_t}$$

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T}\alpha_t h_t(x)\right)$$

Figure 3 shows the boosted FMM framework. When the training accuracy of FMM (output results) drops below 100%, boosted FMM will be activated. A series of training cycles is conducted using AdaBoost to form a new training set. The weights (probability) of incorrectly classified examples by FMM are increased while the weights (probability) of the correctly classified examples are decreased. AdaBoost randomly selects a new training data based on the established probability distribution. The new training set is fed to FMM again for training, and network testing is conducted using the same test set. This process continues until training accuracy achieves 100% or until a pre-specified number of boosting rounds has been reached. The hypotheses from each round are combined to reach a final hypothesis. In the following case studies, we set five rounds of boosting in each experiment.

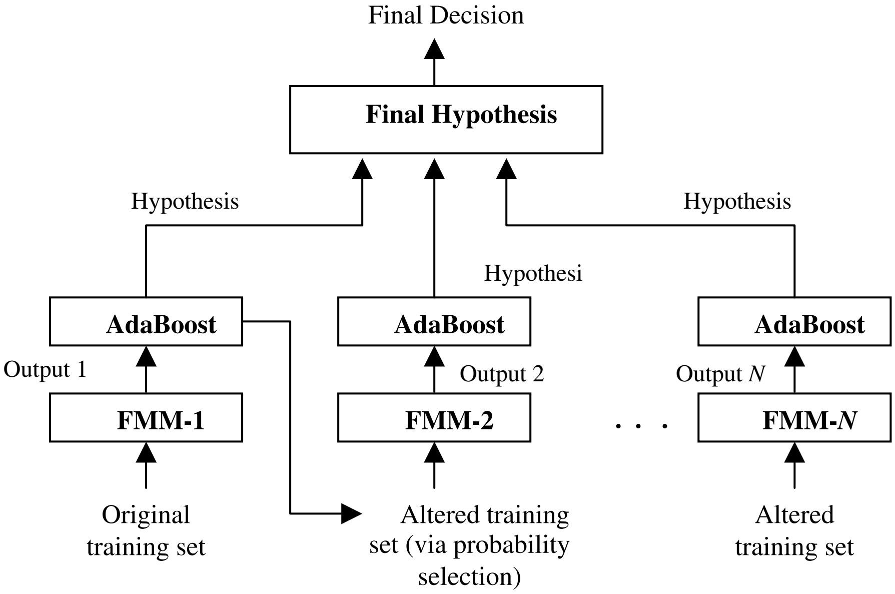


Fig. 3. Procedure for boosting the performance of FMM

2.2.1 Bootstrapping

Although the number of votes associated with the final output can give a confidence estimate of the prediction, we have employed a statistical approach, *i.e.*, the bootstrap method, to estimate the confidence intervals associated with the performance metrics. Bootstrapping is a method for estimating statistical variations of a parameter in situations where the underlying sampling distribution of the parameter is unknown or difficult to estimate. The bootstrap method has been applied to many problems that would be too complicated for traditional statistical analysis to handle, especially when the sample size is small (Efron, 1979, 1981).

With bootstrapping, the observations of the unknown parameter distribution are randomly re-assigned, and the estimates re-calculated. These re-assignments and re-calculations can be done as many times as the experimenter wants and are treated as repeated experiments. In this study, the bootstrap method was implemented to compute the confidence intervals for the performance metrics.

The principle of bootstrapping for computing confidence intervals of a parameter is illustrated as follows. Given a specified random variable $R(\boldsymbol{X}, F)$, possibly depending on both $\boldsymbol{X}$ and the unknown distribution F, estimate the sampling distribution of R on the basis of the observed data:

1. A set of data $\boldsymbol{X} = x_1, \cdots, x_n$ is collected. Suppose that n is the size of the sample observed from a completely unspecified probability distribution F, $\hat{\mu}$ is the mean of all the values in $\boldsymbol{X}$, and N is the number of bootstrap repetitions.
2. Draw a random sample of n data points independently, with replacement, from $\boldsymbol{X}$. Sampling with replacement means that every data point is returned to the data set after use. So, some data points will be included more than once and some will be omitted. The new set of data, $\boldsymbol{X}^*$, is the "bootstrap sample".
3. The bootstrap sample mean of $\boldsymbol{X}^*$, $\hat{\mu}^*$, is calculated.
4. Steps 2 and 3 are repeated N times to obtain N bootstrap estimates of $\hat{\mu}_1^*, \cdots, \hat{\mu}_N^*$.
5. After N bootstrap re-samplings, re-arrange $\hat{\mu}_1^*, \cdots, \hat{\mu}_N^*$ in ascending order, *i.e.* $\hat{\mu}_1^* \le \hat{\mu}_2^* \le \cdots \le \hat{\mu}_N^*$. The $100(1-\alpha)\%$ bootstrap confidence interval is

$$\left(\hat{\mu}_{(q_1)}^*, \hat{\mu}_{(q_2)}^*\right) \tag{6}$$

6. where $q_1 = \left[N\alpha/2\right]$, $q_2 = N - q_1 + 1$, and $1-\alpha$ is the confidence level (*e.g.* to obtain 95% confidence intervals, set $\alpha = 0.05$).

By re-sampling from X many times and computing $\hat{\mu}$ for each of these re-samples, a bootstrapped distribution for $\hat{\mu}$ can be obtained that will approximate the actual distribution of μ. From this, a confidence interval for μ is derived, as described above.

3 Case Studies

3.1 Wisconsin Breast Cancer Data

The Wisconsin Breast Cancer data set was obtained from the UCI repository of Machine Learning Databases[1]. All missing values in the data set were replaced by 0. During the experiment, 2/3 of the data samples were used for training and the remainder for testing. Table 1 shows the FMM results (average of 15 runs with θ = 0.09). Also shown in Table 1 are the results (average of 5 runs) from a number of supervised machine learning algorithms as reported in Eklund and Hoang (2002).

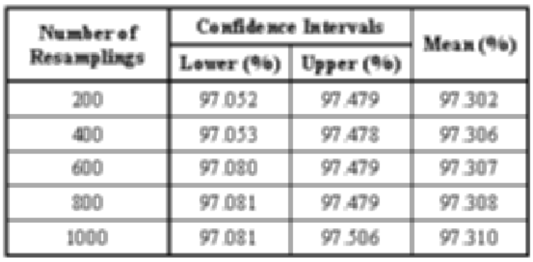
Table 1. Comparison of classification results using the Wisconsin Breast Cancer data set. The results of C4.5, ITI (Incremental Decision Tree Induction), LMDT (Linear Machine Decision Tree) and CN2 are extracted from Eklund and Hoang (2002).

	FMM	C4.5	C4.5 rules	ITI	LMDT	CN2
Test Accuracy (%)	96.74	94.25	94.68	91.14	95.74	94.39
Standard Deviation	0.3914	2.36	2.40	6.79	3.03	3.60

In order to ascertain the stability of the FMM's performance, bootstrapping was applied to determine the 95% confidence intervals of the accuracy rate. The results are shown in Table 2. Notice that the difference between the lower and the upper limits of the confidence intervals was small, suggesting that the network performance was stable.

[1] **http://www.ics.uci.edu/~mlearn/MLRepository.html**

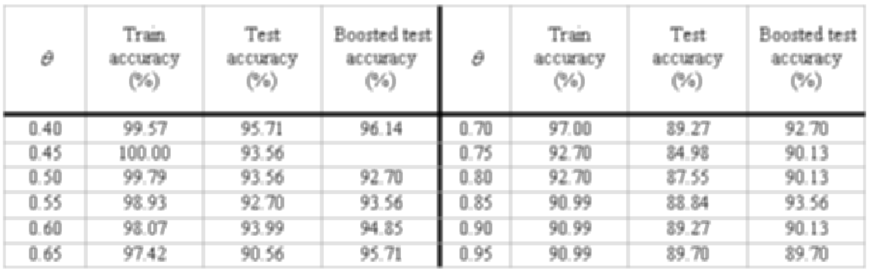
Table 2. Bootstrapped results with different number of resamplings



Number of Resamplings	Confidence Intervals		Mean (%)
	Lower (%)	Upper (%)	
200	97.052	97.479	97.302
400	97.053	97.478	97.306
600	97.080	97.479	97.307
800	97.081	97.479	97.308
1000	97.081	97.506	97.310

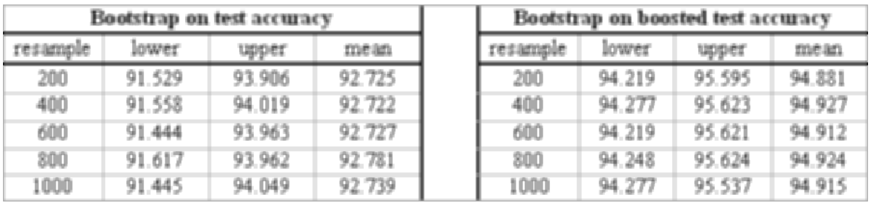
When the training accuracy rate dropped below 100%, AdaBoost was used to boost the FMM performance. Table 3 shows the accuracy rates of training, testing, and boosted results obtained by different settings of θ. As the results for $\theta < 0.40$ and $\theta = 0.45$ reached 100%, AdaBoost was not applied. Note that when $\theta = 0.50$, the boosted test accuracy rate (92.70%) is lower than the non-boosted one (93.56%). This might result from near-perfect training accuracy (99.79%). Further investigations were conducted to determine the stability of these results by using bootstrapping. As shown in Table 4(a), the lower bound of the 95% confidence intervals of boosted FMM results were higher than the upper bound of the non-boosted FMM results, suggesting boosted FMM performed significantly better than non-boosted FMM. Bootstrapping was applied to the results for $\theta = 0.65$, as summarized in Table 4(b). The outcomes again showed that boosted FMM performed better that non-boosted FMM.

Table 3. Training, testing and boosted test accuracy rates of FMM. When the training accuracy rate achieves 100%, boosting is not applicable.



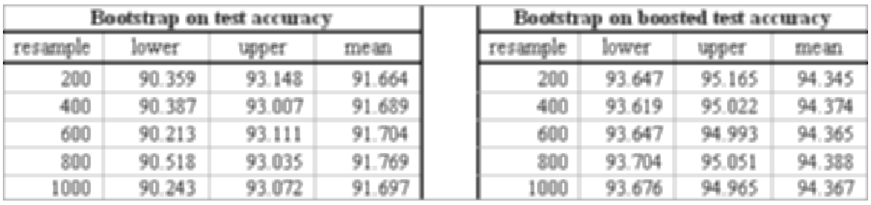
θ	Train accuracy (%)	Test accuracy (%)	Boosted test accuracy (%)	θ	Train accuracy (%)	Test accuracy (%)	Boosted test accuracy (%)
0.40	99.57	95.71	96.14	0.70	97.00	89.27	92.70
0.45	100.00	93.56		0.75	92.70	84.98	90.13
0.50	99.79	93.56	92.70	0.80	92.70	87.55	90.13
0.55	98.93	92.70	93.56	0.85	90.99	88.84	93.56
0.60	98.07	93.99	94.85	0.90	90.99	89.27	90.13
0.65	97.42	90.56	95.71	0.95	90.99	89.70	89.70

Table 4(a). Bootstrapped results with different number of resamplings for $\theta = 0.50$



Bootstrap on test accuracy				Bootstrap on boosted test accuracy			
resample	lower	upper	mean	resample	lower	upper	mean
200	91.529	93.906	92.725	200	94.219	95.595	94.881
400	91.558	94.019	92.722	400	94.277	95.623	94.927
600	91.444	93.963	92.727	600	94.219	95.621	94.912
800	91.617	93.962	92.781	800	94.248	95.624	94.924
1000	91.445	94.049	92.739	1000	94.277	95.537	94.915

Table 4(b). Bootstrapped results with different number of resamplings for $\theta = 0.65$



Bootstrap on test accuracy				Bootstrap on boosted test accuracy			
resample	lower	upper	mean	resample	lower	upper	mean
200	90.359	93.148	91.664	200	93.647	95.165	94.345
400	90.387	93.007	91.689	400	93.619	95.022	94.374
600	90.213	93.111	91.704	600	93.647	94.993	94.365
800	90.518	93.035	91.769	800	93.704	95.051	94.388
1000	90.243	93.072	91.697	1000	93.676	94.965	94.367

3.2 Pima Indians Diabetes Data

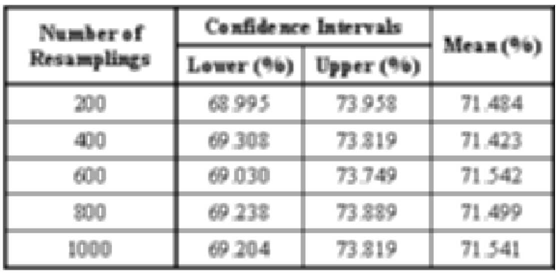
The Pima Indians Diabetes was also obtained from the UCI repository of Machine Learning Databases. It contained 768 samples with no missing values. The samples were divided into a training set and a test set, each comprising 2/3 and 1/3 of the total number of samples. Table 5 shows the FMM results (average of 15 runs with $\theta = 0.40$), together with those from other algorithms as reported in Eklund and Hoang (2002).

Table 5. Comparison of classification results using the Pima Indians Diabetes data set. The results of C4.5, ITI (Incremental Decision Tree Induction), LMDT (Linear Machine Decision Tree) and CN2 are extracted from Eklund and Hoang (2002).

	FMM	C4.5	C4.5 rules	ITI	LMDT	CN2
Test Accuracy (%)	71.49	71.02	71.05	73.16	73.51	72.19
Standard Deviation	4.736	2.10	3.92	2.16	4.30	2.36

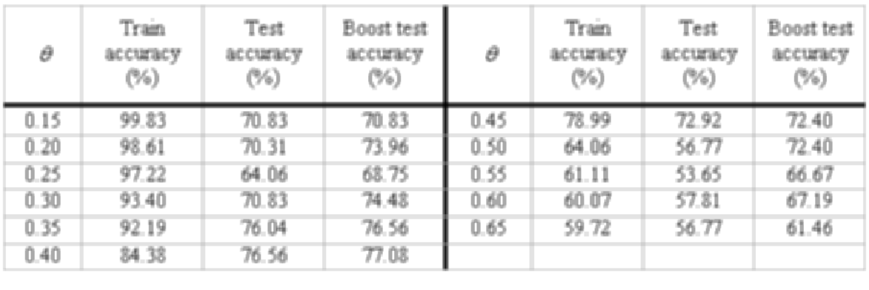
Bootstrapping was applied to the results to determine the 95% confidence intervals of the network performance. The results are shown in Table 6. Notice that the difference between the lower and upper limits of the confidence interval was small, again indicating that the network performance was stable. As in the previous experiment, AdaBoost was implemented to boost the performance of FMM. Table 7 shows the accuracy rates of training, testing and boosted results obtained by different settings of θ. AdaBoost is not applied to FMM when $\theta < 0.15$ owing to perfect score being achieved. Notice that when the training accuracy rates dropped to less than 50% ($\theta > 0.65$), AdaBoost failed to produce any results (see the procedure of AdaBoost for details). When $\theta = 0.45$, the boosted test accuracy (72.40%) was slightly lower than that of the non-boosted test accuracy (72.92%). Bootstrapping was applied to determine the 95% confidence intervals of these results. Table 8(a) summarizes the outcome. The results indicate that the mean accuracy rates of boosted FMM were better than those from non-boosted FMM. Bootstrapping was again applied to the results from $\theta = 0.50$, as shown in Table 8(b). The same finding can be observed, i.e., boosted FMM performed better than non-boosted FMM. Overall, the highest accuracy achieved using boosted FMM was 77.08% with $\theta = 0.40$, as shown in Table 7.

Table 6. Bootstrapped results with different number of resamplings



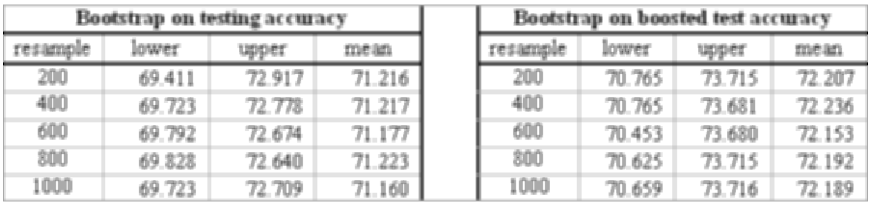
Number of Resamplings	Confidence Intervals		Mean (%)
	Lower (%)	Upper (%)	
200	68.995	73.958	71.484
400	69.308	73.819	71.423
600	69.030	73.749	71.542
800	69.238	73.889	71.499
1000	69.204	73.819	71.541

Table 7. Training, testing and boosted test accuracy rates of FMM



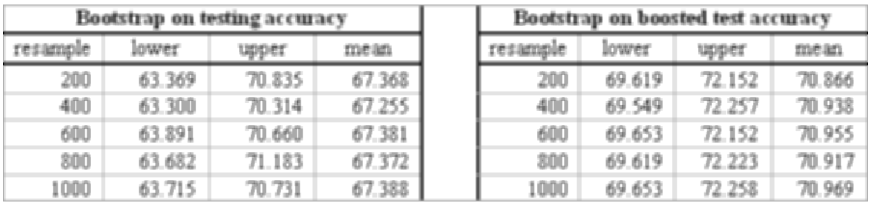
θ	Train accuracy (%)	Test accuracy (%)	Boost test accuracy (%)	θ	Train accuracy (%)	Test accuracy (%)	Boost test accuracy (%)
0.15	99.83	70.83	70.83	0.45	78.99	72.92	72.40
0.20	98.61	70.31	73.96	0.50	64.06	56.77	72.40
0.25	97.22	64.06	68.75	0.55	61.11	53.65	66.67
0.30	93.40	70.83	74.48	0.60	60.07	57.81	67.19
0.35	92.19	76.04	76.56	0.65	59.72	56.77	61.46
0.40	84.38	76.56	77.08				

Table 8(a). Bootstrapped results with different number of resamplings when $\theta =$ 0.45



Bootstrap on testing accuracy					Bootstrap on boosted test accuracy			
resample	lower	upper	mean		resample	lower	upper	mean
200	69.411	72.917	71.216		200	70.765	73.715	72.207
400	69.723	72.778	71.217		400	70.765	73.681	72.236
600	69.792	72.674	71.177		600	70.453	73.680	72.153
800	69.828	72.640	71.223		800	70.625	73.715	72.192
1000	69.723	72.709	71.160		1000	70.659	73.716	72.189

Table 8(b). Bootstrapped results with different number of resamplings when $\theta =$ 0.50



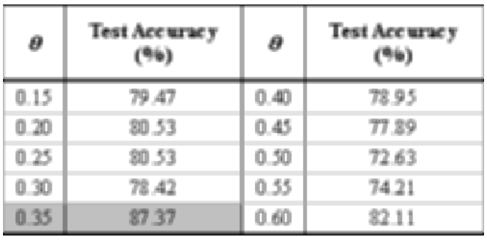
Bootstrap on testing accuracy					Bootstrap on boosted test accuracy			
resample	lower	upper	mean		resample	lower	upper	mean
200	63.369	70.835	67.368		200	69.619	72.152	70.866
400	63.300	70.314	67.255		400	69.549	72.257	70.938
600	63.891	70.660	67.381		600	69.653	72.152	70.955
800	63.682	71.183	67.372		800	69.619	72.223	70.917
1000	63.715	70.731	67.388		1000	69.653	72.258	70.969

3.3 Myocardial Infraction (MI) Data

Myocardial Infraction (commonly known as heart attack) is one of the major causes of mortality in the developed world. This disease is usually difficult to identify in the early stage because the symptoms are not always typical and may arise from a variety of other conditions that are not necessarily related to MI. Therefore, it is important to have a tool for rapid diagnosis of MI so that early treatment can be offered to suspected MI patients. This motivates the use of FMM as a decision support tool for the early diagnosis of MI.

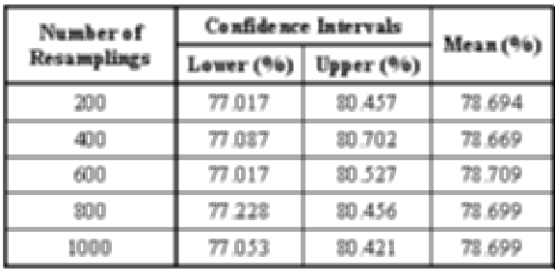
The data set was obtained from the Northern General Hospital, Sheffield, UK. There were altogether 474 patient records (data samples) in the data set. The samples were divided into a training set and a test set, each comprising 2/3 and 1/3 of the total number of samples. After several trial runs, FMM produced the best test results when θ was varied from 0.10 to 0.60. As shown in Table 9, the best accuracy rate achieved was 87.37%. Bootstrapping was applied to the results to determine the 95% confidence intervals of the network performance. The results are shown in Table 10. Again, the small confidence bounds indicate that the network performance was stable.

Table 9. Test results from varying θ



θ	Test Accuracy (%)	θ	Test Accuracy (%)
0.15	79.47	0.40	78.95
0.20	80.53	0.45	77.89
0.25	80.53	0.50	72.63
0.30	78.42	0.55	74.21
0.35	87.37	0.60	82.11

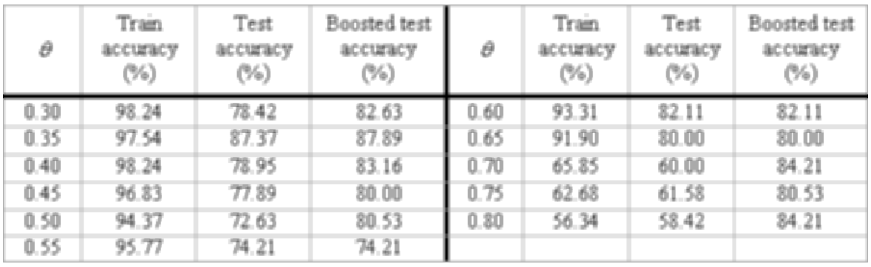
Table 10. Bootstrapped results



Number of Resamplings	Confidence Intervals		Mean (%)
	Lower (%)	Upper (%)	
200	77.017	80.457	78.694
400	77.087	80.702	78.669
600	77.017	80.527	78.709
800	77.228	80.456	78.699
1000	77.053	80.421	78.699

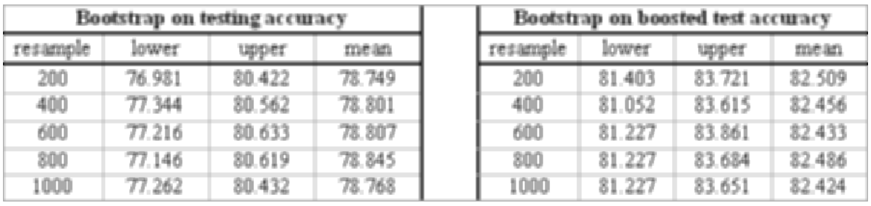
Table 11 shows the accuracy rates of training, testing and boosted results obtained by different settings of θ. As in previous experiments, further examination with bootstrapping (using results from $\theta = 0.35$) shows that boosted FMM performed better than non-boosted FMM, as shown in Table 12(a). Table 12(b), again, confirms that boosted FMM is able to achieve results that are statistically better that non-boosted FMM.

Table 11. Training, testing and boosted test accuracy rates of FMM



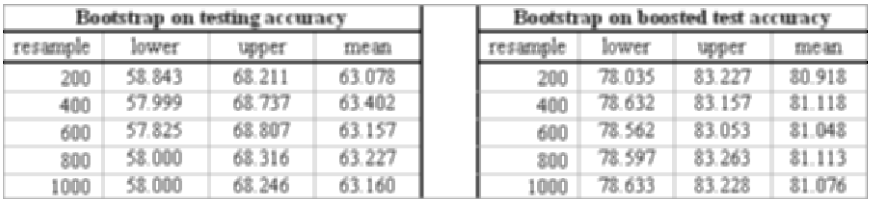
θ	Train accuracy (%)	Test accuracy (%)	Boosted test accuracy (%)	θ	Train accuracy (%)	Test accuracy (%)	Boosted test accuracy (%)
0.30	98.24	78.42	82.63	0.60	93.31	82.11	82.11
0.35	97.54	87.37	87.89	0.65	91.90	80.00	80.00
0.40	98.24	78.95	83.16	0.70	65.85	60.00	84.21
0.45	96.83	77.89	80.00	0.75	62.68	61.58	80.53
0.50	94.37	72.63	80.53	0.80	56.34	58.42	84.21
0.55	95.77	74.21	74.21				

Table 12(a). Bootstrapped results with different number of resamplings for $\theta =$ 0.35



Bootstrap on testing accuracy					Bootstrap on boosted test accuracy			
resample	lower	upper	mean		resample	lower	upper	mean
200	76.981	80.422	78.749		200	81.403	83.721	82.509
400	77.344	80.562	78.801		400	81.052	83.615	82.456
600	77.216	80.633	78.807		600	81.227	83.861	82.433
800	77.146	80.619	78.845		800	81.227	83.684	82.486
1000	77.262	80.432	78.768		1000	81.227	83.651	82.424

Table 12(b). Bootstrapped results with different number of resamplings for $\theta =$ 0.70



Bootstrap on testing accuracy					Bootstrap on boosted test accuracy			
resample	lower	upper	mean		resample	lower	upper	mean
200	58.843	68.211	63.078		200	78.035	83.227	80.918
400	57.999	68.737	63.402		400	78.632	83.157	81.118
600	57.825	68.807	63.157		600	78.562	83.053	81.048
800	58.000	68.316	63.227		800	78.597	83.263	81.113
1000	58.000	68.246	63.160		1000	78.633	83.228	81.076

4 Conclusions

In this paper, we investigate the use of AdaBoost in conjunction with FMM in an attempt to improve the classification accuracy of the FMM network. The performance of the FMM and boosted FMM networks were evaluated using two benchmark studies and a real medical diagnosis problem. The results consistently demonstrate that boosted FMM performed better, statistically, than non-boosted FMM. Further work will concentrate on the evaluation and validation of boosted FMM as a practical pattern classification tool for handling a variety of real-world problems.

References

Breiman, L. (1996), *Bias, variance, and arcing classifiers*. Technical Report 460, Statistics Department, University of California at Berkeley.

Devijver, P., and Kittler, P. (1982), *Pattern Recognition: A Statistical Approach*. Englewood Cliffs, NJ: Prentice-Hall.

Duda, R.O., and Hart, P.E. (1973), *Pattern Classification and Scene Analysis*. New York: Wiley.

Efron, B. (1979), "Bootstrap Methods: Another Look at the Jackknife", *The Annals Of Statistics*, vol. 7, pp. 1-26.

Efron, B. (1981), "Nonparametric Standard Errors and Confidence Intervals", *The Canadian Journal of Statistics*, 9, 139-172.

Eklund, P. W., and Hoang, A. (2002), *A Performance Survey of Public Domain Supervised Machine Learning Algorithms*, Tech. Reports, School of Information Technology, Griffith University, Queensland.

Freund, Y., and Schapire, R.E. (1996), "Experimentals with a new boosting algorithm," In *Machine Learning: Proceedings of Thirteenth International Conference*, pages 148-156.

Freund, Y., and Schapire, R.E. (1997), "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Science*, 55(1): 119-139.

Freund, Y., and Schapire, R.E. (1999), "A short introduction to boosting," *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780.

Fukunaga, K. (1972), *Introduction to Statistical Pattern Recognition*. New York: Academic Press.

Kearns, M., and Valiant, L.G. (1988), *Learning Boolean formulae or finite automata is as hard as factoring*, Technical Report TR-14-88, Harvard University Aiken Computation Laboratory.

Kearns, M., and Valiant, L.G. (1994), "Cryptographic limitations on learning Boolean formulae and finite automata," *Journal of the Association fro Computing Machinery*, 41(1):67-95.

Schapire, R.E. (1990), "The strength of weak learnability," *Machine Leaning*, 5(2):197-227.

Simpson, P. (1992), "Fuzzy Min-Max Neural Networks – Part 1: Classification", *IEEE Transaction on Neural Networks*, vol. 3, pp 776-786.

Valiant, L.G. (1984), "A theory of the learnable," *Communication of the ACM*, 27(11):1134-1142.

A Genetic Algorithm for Solving BSS-ICA

J.M. Górriz[1] and C.G. Puntonet[2]

[1] Dpt. of Signal Theory, University of Granada , Spain gorriz@ugr.es
[2] Dpt. Architecture and Computer Tech. University of Granada, Spain carlos@atc.ugr.es

In this paper we proposed a genetic algorithm to minimize a nonconvex and nonlinear cost function based on statistical estimators for solving blind source separation-independent component analysis problem. In this way a novel method for blindly separating unobservable independent component signals from their linear and non linear (using mapping functions) mixtures is devised. The GA presented in this work is able to extract independent components with faster rate than the previous independent component analysis algorithms based on Higher Order Statistics (HOS) as input space dimension increases showing significant accuracy and robustness.

1 Introduction

The starting point in the Independent Component Analysis (ICA) research can be found in (Barlow et al. 1961) where a principle of redundancy reduction as a coding strategy in neurons was suggested, i.e. each neural unit was supposed to encode statistically independent features over a set of inputs. But it was in the 90´s when Bell and Sejnowski applied this theoretical concept to the blindly separation of the mixed sources (BSS) using a well known stochastic gradient learning rule (Bell et al. 1995) and originating a productive period of research in this area (Hyvarinen et al. 1997). In this way ICA algorithms have been applied successfully to several fields such as biomedicine, speech, sonar and radar, signal processing, etc. and more recently also to time series forecasting (Górriz 2003), i.e. using stock data.

On the other hand there is a wide class of interesting applications for which no reasonably fast algorithms have been developed, i.e. optimization problems that appear frequently in several applications such as VLSI design or the travelling salesman problem. In general, any abstract task to be accomplished can be viewed as a search through a space of potential solutions and whenever we work with large spaces, GAs are suitable artificial intelligence techniques for developing this optimization (Michalewicz et al. 1992). In this work we apply GA to ICA in the search of the separation matrix, in order to improve the performance of endogenous learning machines in real time series forecasting speeding up convergence rates (scenarios with the BSS problem in higher dimension). We organize the essay as follows. In section 2 we give a brief overview of the basic GA theory and introduce a set of new genetic operators in sections 3 and 6. In the next sections we derived a suitable fitness function for the GA used and finally the new search algorithm will be compare to the well-known ICA algorithms and state state some conclusions in section 7.

2 Basis Genetic Algorithms

A GA can be modelled by means of a *time inhomogeneous Markov* chain (Haggstromn et al. 1998) obtaining interesting properties related with weak and strong ergodicity, convergence and the distribution probability of the process (Lothar et al. 1998). A canonical GA is constituted by operations of parameter encoding, population initialization, crossover , mutation, mate selection, population replacement, fitness scaling, (see (Górriz 2003),(Lothar et al. 1998) for definitions and properties of the genetic operators) etc. proving that with these simple operators a GA does not converge to a population containing only optimal members. However, there are GAs that converge to the optimum, *The Elitist GA* (Suzuki et al. 1995) and those which introduce *Reduction Operators*(Górriz 2003). We have borrowed the notation mainly from (Lothar et al. 1998) where the model for GAs is a inhomogeneous Markov chain model on probability distributions (**S**) over the set of all possible populations of a fixed finite size. Let **C** the set of all possible creatures in a given world (number of vectors of genes equal to that of elements of the mixing matrix) and a function $f : \mathbf{C} \to R^+$ (see section 5). The task of GAs is to find an element $c \in \mathbf{C}$ for which $f(c)$ is maximal. We encode creatures into genes and chromosomes or individuals as strings of length ℓ of binary digits (size of Alphabet A is $a = 2$) using one-complement representation.

3 Mutation Operator based on neighborhood philosophy

The new Mutation Operator $\mathbf{M}_{\mathbf{P_m}}$ is applied (with probability $\mathbf{P}_m$) independently at each bit in a population $p \in \wp_N$, to avoid premature convergence (see (Michalewicz et al. 1992) for further discussion) and enforcing strong ergodicity. The multi-bit mutation operator with change probability following a *exponential* law with respect to the position $1 \leq i \leq L$ in $p \in \wp_N$:

$$P_m(i) = \mu \cdot \exp\left(\frac{-mod\{\frac{i-1}{N}\}}{\emptyset}\right) \tag{1}$$

where $\emptyset$ is a normalization constant and μ the change probability at the beginning of each creature p_i in population p; can be described as a positive stochastic matrix (see (Górriz 2003)). This operator has similar properties to the Constant Multiple-bit mutation operator $\mathbf{M}_\mu$ (Lothar et al. 1998). $\mathbf{M}_\mu$ is a contracting map in the sense presented in (Lothar et al. 1998). It is easy to prove that $\mathbf{M}_{\mathbf{P_m}}$ is a contracting map too, using the eigenvalues of this operator. We can also compare the coefficients of ergodicity:

$$\tau_r(\mathbf{M}_{\mathbf{P_m}}) < \tau_r(\mathbf{M}_\mu) \tag{2}$$

where $\tau_r(\mathbf{X}) = max\{\|\mathbf{X}v\|_r : \quad v \in \mathcal{R}^n, \quad v \perp e \quad and \quad \|v\|_r = 1\}$.

Mutation is more likely at the beginning of the string of binary digits ("small neighborhood philosophy"). In order to improve the speed convergence of the algorithm we have included mechanisms such as elitist strategy (reduction operator (Lozano et al. 1999) consisting of sampling a Boltzmann probability distribution in the extended population) in which the best individual in the current generation always survives into the next (a further discussion about reduction operator, $\mathbf{P}_R$, can be found in (Rudolph et al. 1994)).

4 ICA and Convex Optimization under Discrepancy Constraints

In order to solve ICA problems using the SVM paradigm, we use an approach that is based on reformulating the determination of the unknown un-mixing matrix $\mathbf{A}$ as a convex optimization problem. The optimization program we are about to formulate is later solved using the Lagrange multipliers method coupled with an approximation to a given derivative of a convenient discrepancy function based on cumulants or on the characteristic function of the original sources. Note that our approach could also be easily modified to take into account other paradigms in ICA research such as density estimation-based approximation methods.

We first restrict the range of possible solutions to the problem, by what usually is a reasonable normalizing constraint: that the Frobenius norm of the sought matrix $\mathbf{A}$ be minimum. We take this, however, to be our explicit objective function:

$$minimize \quad \frac{1}{2} \cdot \|\mathbf{A}\|_2^2, subject\ to \quad \tilde{L}(\mathbf{a}_i) < \epsilon \ \ , (i = 1, 2, \ldots, n) \ . \tag{3}$$

because it makes our program a convex programming one (at least with the Frobenius norm). The discrepancy between the model and what one iteratively observes will be contained in the restrictions in equation 3. For each time instant t, we have $\tilde{L}(\mathbf{a}_i) \approx < \mathbf{a}_i, \mathbf{x} > -c_i - s_i$, with $\mathbf{a}_i$ denoting the i-th row of un-mixing matrix $\mathbf{A}$, and c_i being the i-th component on vector $\mathbf{c}$. Note that for simplicity we have not written the dependency on the time instant t, but of course we have to take that into account when implementing.

We define the Lagrangian corresponding to (3) as

$$\mathcal{L}_i = \frac{1}{2} \cdot \|\mathbf{a}_i\|_2^2 + C \cdot \sum_{j=1}^{l} (\xi_j + \xi_j^*) - \sum_{j=1}^{l} \alpha_j (\epsilon + \xi_j + \tilde{L}(\mathbf{a}_i) - \tag{4}$$

$$- \sum_{j=1}^{l} \alpha_j^* (\epsilon + \xi_j^* - \tilde{L}) - \sum_{j=1}^{l} (\eta_j \xi_j + \eta_j^* \xi_j^*) \ . \tag{5}$$

Now we take the corresponding partial derivatives (according to the Lagrangian method) and equal them to 0, as follows

$$\partial_b \mathcal{L}_i = \sum_{j=1}^{l} (\alpha_j^* + \alpha_j) = 0 \ . \qquad \partial_{\theta_i^*} \mathcal{L}_i = C - \alpha_j^* - \eta_j^* = 0 \ . \tag{6}$$

$$\partial_{\mathbf{a}_i} \mathcal{L}_i = \mathbf{a}_i - \sum_{j=1}^{l} (\alpha_j - \alpha_j^*) \cdot \partial_{\mathbf{a}_i} \tilde{L}(\mathbf{a}_i) = 0 \ . \tag{7}$$

Attending to equation 7 we can notice how the algorithm can extract independent components one by one just working with the maximization of the selected Lagrangian function $\mathcal{L}_i$. The selection of a suitable function $\tilde{L}(\mathbf{a}_i, \mathbf{x})$ determines the current algorithm or strategy used in process, i.e. if we describe it in terms of neg-entropy we will obtain a generalization of FastICA (Hyvarinen et al. 1997). After a bit of algebraic manipulation, we get to

$$\mathcal{L}_i = \frac{1}{2} \cdot \left\| \sum_{j=1}^{l} (\alpha_j - \alpha_j^*) \partial_{\mathbf{a}_i} \tilde{L}(\mathbf{a}_i) \right\|^2 - \epsilon \sum_{j=1}^{l} (\alpha_j + \alpha_j^*) - \sum_{j=1}^{l} (\alpha_j - \alpha_j^*) \tilde{L}(\mathbf{a}_i) \ . \tag{8}$$

Finally we transform ICA into a multidimensional maximization of the Lagrangian function defined as:

$$\mathcal{L} = (\mathcal{L}_1 \mathcal{L}_2, \ldots, \mathcal{L}_n) \tag{9}$$

5 A New Statistical Independence Criterion: the Search for a Suitable Fitness Function

The Statistical Independence of a set of random variables can be described in terms of their joint and individual probability distribution. The independence condition for the independent components of the output vector $\mathbf{y}$ is given by the definition of independence random variables:

$$p_{\mathbf{y}}(\mathbf{y}) = \prod_{i=1}^{n} p_{\mathbf{y}_i}(y_i) \tag{10}$$

where $p_{\mathbf{y}}$ is the joint pdf of random vector (observed signals) $\mathbf{y}$ and $p_{\mathbf{y}_i}$ is the marginal pdf of y_i. In order to measure the independence of the outputs we express equation 10 in terms of higher order statistics (cumulants) using the characteristic function (or moment generating function) $\phi(\mathbf{k})$, where $\mathbf{k}$ is a vector of variables in the Fourier transform domain, and considering its natural logarithm $\Phi = log(\phi(\mathbf{k}))$. We first evaluate the difference between the terms in equation 10 to get:

$$\pi(\mathbf{y}) = \left\| p_{\mathbf{y}}(\mathbf{y}) - \prod_{i=1}^{n} p_{\mathbf{y}}(y_i) \right\|^2 \tag{11}$$

where the norm $||\dots||^2$ can be defined using the convolution operator with different window functions according to the specific application (Tan et al. 2001) as follows:

$$\|F(y)\|^2 = \int \{F(\mathbf{y}) * v(\mathbf{y})\}^2 d\mathbf{y} \tag{12}$$

and $v(\mathbf{y}) = \prod_{i=1}^{n} w(y_i)$. In the Fourier domain and taking natural log (in order to use higher order statistics, i.e. cumulants) this equation transforms into:

$$\Pi(\mathbf{k}) = \int \left\| \mathbf{\Psi}_{\mathbf{y}}(\mathbf{k}) - \sum_{i=1}^{n} \mathbf{\Psi}_{\mathbf{y_i}}(k_i) \right\|^2 \mathbf{V}(\mathbf{k}) d\mathbf{k} \tag{13}$$

where $\mathbf{\Psi}$ is the cumulant generating or characteristic function (the natural log of the moment generating function) y V is the Fourier transform of the selected window function $v(\mathbf{y})$. If we take Taylor expansion around the origin of the characteristic function we obtain:

$$\mathbf{\Psi}_{\mathbf{y}}(\mathbf{k}) = \sum_{\lambda} \frac{1}{\lambda!} \frac{\partial^{|\lambda|} \mathbf{\Psi}_{\mathbf{y}}}{\partial k_1^{\lambda_1} \dots \partial k_n^{\lambda_n}} (\mathbf{0}) k_1^{\lambda_1} \dots k_n^{\lambda_n} \tag{14}$$

where we define $|\lambda| \equiv \lambda_1 + \ldots + \lambda_n$, $\lambda \equiv \{\lambda_1 \ldots \lambda_n\}$, $\lambda! \equiv \lambda_1! \ldots \lambda_n!$ and:

$$\mathbf{\Psi}_{\mathbf{y_i}}(\mathbf{k_i}) = \sum_{\lambda_i} \frac{1}{\lambda_i!} \frac{\partial^{\lambda_i} \mathbf{\Psi}_{\mathbf{y_i}}}{\partial k_i^{\lambda_i}} (0) k_i^{\lambda_i} \tag{15}$$

where the factors in the latter expansions are the cumulants of the outputs (cross and not cross cumulants):

$$C_{y_1 \dots y_n}^{\lambda_1 \dots \lambda_n} = (-j)^{|\lambda|} \frac{\partial^{\lambda_1 + \ldots + \lambda_n} \mathbf{\Psi}_{\mathbf{y}}}{\partial k_1^{\lambda_1} \dots \partial k_n^{\lambda_n}} (\mathbf{0}) \qquad C_{y_i}^{\lambda_i} = (-j)^{\lambda_i} \frac{\partial^{\lambda_i} \mathbf{\Psi}_{\mathbf{y_i}}}{\partial k_i^{\lambda_i}} (0) \tag{16}$$

Thus if we define the difference of the terms in equation 13 as:

$$\beta_\lambda = \frac{1}{\lambda!}(j)^{|\lambda|}C_{\mathbf{y}}^{\lambda} \tag{17}$$

that contains the infinite set of cumulants of the output vector $\mathbf{y}$. Substituting 17 into 13 we have:

$$\Pi(\mathbf{k}) = \int \left\| \sum_{\lambda} \beta_\lambda k_1^{\lambda_1} \dots k_n^{\lambda_n} \right\|^2 \mathbf{V}(\mathbf{k})d\mathbf{k} \tag{18}$$

Hence vanishing cross-cumulants are a necessary condition for $y_1, \dots, y_n$ to be independent[3]. Equation 18 can be transformed interchanging the sequence of summation and integral into:

$$\Pi = \sum_{\lambda,\lambda^*} \beta_\lambda \beta^*_{\lambda^*} \mathbf{\Gamma}_{\lambda,\lambda^*} \tag{19}$$

where $\mathbf{\Gamma} = \int k_1^{\lambda_1+\lambda_1^*} \dots k_n^{\lambda_n+\lambda_n^*}\mathbf{V}(\mathbf{k})d\mathbf{k}$. In this way we finally describe the generic function $\tilde{\mathbf{L}}$ in the Lagrangian function $\mathcal{L}$. We must impose some additional restrictions to $\tilde{\mathbf{L}}$, it will be a version of the previous one but limiting the set λ. That is, we only consider a finite set of cumulants $\{\lambda, \lambda^*\}$ such us $|\lambda| + |\lambda^*| < \tilde{\lambda}$ and include only the cumulants affecting to the current Lagrangian component. Mathematically we can express these two restriction as:

$$\tilde{\mathbf{L}}_i \equiv \Pi(\theta) = \sum_{\{\lambda,\lambda^*\}} \beta_\lambda \beta^*_{\lambda^*} \mathbf{\Gamma}_{\lambda,\lambda^*} \qquad \backslash \left\{ \begin{array}{c} \{\lambda,\lambda^*\} \bigcap \{\lambda_i\} \neq 0 \\ |\lambda| + |\lambda^*| < \tilde{\lambda} \end{array} \right\} \tag{20}$$

where the set of paramenters encoded by the GA consist on the elements of the separation matrix and the lagrange multipliers $\theta = \{a_{ij}, \alpha_i, \alpha_i^*\}$.

In order to evaluate the most relevant term in the Lagrangian $\frac{\partial \tilde{\mathbf{L}}}{\partial \mathbf{a_i}}$ we have to rewrite the above-written equations in terms of the output vector as $y_i = \mathbf{a_i x}$ and to use the connection between cumulants and moments shown in (Chryssosto-mos et al. 1993):

$$C_{\mathbf{y}}^{\lambda} = \sum_{p_1,\dots,p_m} (-1)^{m-1}(m-1)! \cdot E[\prod_{j\in p_1} Y_j] \dots E[\prod_{j\in p_m} Y_j] \tag{21}$$

where $\{p_1, \dots, p_m\}$ are all the possible partitions with $m = 1, \dots, \lambda$ included in the set of integers $\{1, \dots, \lambda\}$. In SVM methodology we work with instantaneous values (sample by sample) thus we have to approximate expected value to instantaneous ones. Finally evaluating the derivative term in equation 21 and using the above-mentioned approximations we reach to:

$$\frac{\partial C_{\mathbf{y}}^{\lambda}}{\partial \mathbf{a_i}} = \sum_{p_1,\dots,p_m} (-1)^{m-1}(m-1)! \cdot \sum_{k=1}^{m} \left(\frac{s_k (A^{-1}\cdot \mathbf{y})^{s_k - 1}}{y_i^{s_k}} \prod_{j\in p_1} y_j \cdots \prod_{j\in p_m} y_j \right) \tag{22}$$

[3] In practice we need independence between sources two against two.

where λ satisfies the conditions shown in equation 20, s_k is an integer in the set $\{1, \ldots, \tilde{\lambda}\}$. In practice the order of the statistics used never exceed to four or five so the latter expression can be simplified significantly, rewriting the cumulants in terms of dot products between the output signals y_i. Expressions of cumulants in terms of moments are well-known in numerous references thus with the equation 22 and equation 7 allow us to iteratively obtain the coefficients α_j, α_j^* and then the support vector parameters $\mathbf{a_i}$ of the separation matrix $\mathbf{A}$:

$$\mathbf{a}_i = \sum_{j=1}^{l} (\alpha_j - \alpha_j^*) \cdot \partial_{\mathbf{a}_i} \tilde{L}(\mathbf{a}_i) = \sum_{j=1}^{l} (\alpha_j - \alpha_j^*) \cdot \sum_{\{\lambda,\lambda^*\}} \partial_{\mathbf{a}_i} \left(\beta_\lambda \beta_{\lambda^*}^*\right) \mathbf{\Gamma}_{\lambda,\lambda^*} \quad (23)$$

$$= \sum_{j=1}^{l} (\alpha_j - \alpha_j^*) \cdot \sum_{\{\lambda,\lambda^*\}} \frac{(j)^{|\lambda|+|\lambda^*|}}{\lambda! \lambda^*!} \partial_{\mathbf{a}_i} \left(\mathbf{C}_{\mathbf{y}}^{\lambda} \mathbf{C}_{\mathbf{y}}^{\lambda^*} \right) \mathbf{\Gamma}_{\lambda,\lambda^*} \quad (24)$$

6 Guided Genetic Algorithm

In order to include statistical information into the algorithm (it would be a nonsense to ignore it!) we define the hybrid statistical genetic operator based on reduction operators as follows:

$$\langle q, \mathbf{M}_{\mathbf{G}}^{\mathbf{n}} p \rangle = \frac{1}{\aleph(T_n)} \exp\left(-\frac{||q - \mathbf{S^n} \cdot p||^2}{T_n} \right); \qquad p, q \in \wp_N \qquad (25)$$

where $\aleph(T_n)$ is the normalization constant depending on temperature T_n, n is the iteration and $\mathbf{S^n}$ is the step matrix which contains statistical properties, i.e based on cumulants it can be expressed using quasi-Newton algorithms as (Hyvarinen et al. 1997):

$$\mathbf{S^n} = (\mathbf{I} - \mu^n (\mathbf{C}_{y,y}^{1,\beta} \mathbf{S}_y^{\beta} - \mathbf{I})); \qquad p_i \in \mathbf{C} \qquad (26)$$

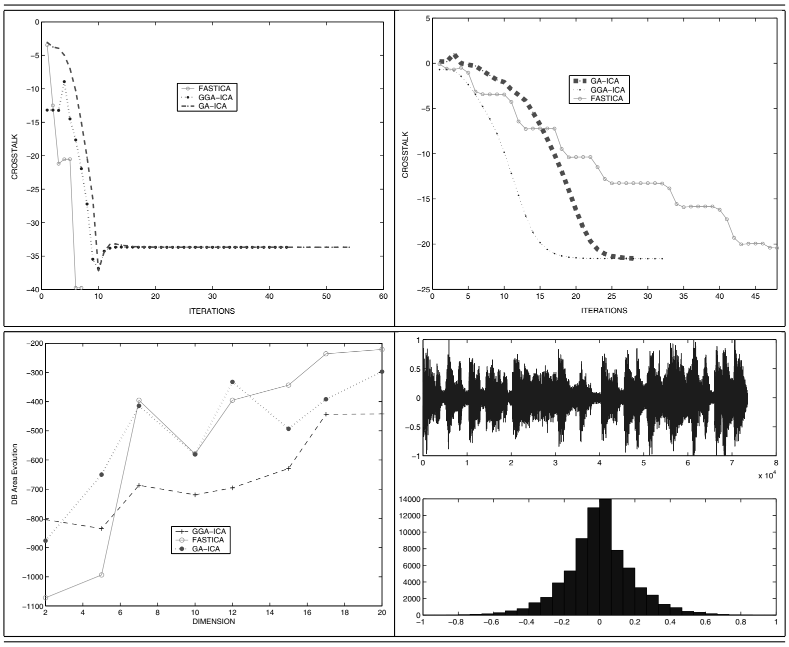
where $\mathbf{C}_{y,y}^{1,\beta}$ is the cross-cumulant matrix whose elements are $[\mathbf{C}_{y,y}^{\alpha,\beta}]_{ij} = Cum(\underbrace{y_i, \ldots, y_i}_{\alpha}, \underbrace{y_j, \ldots, y_j}_{\beta})$ and $\mathbf{S}_y^{\beta}$ is the sign matrix of the output cumulants.

Such search requires balancing two goals: exploiting the blindly search like a canonical GA and using statistical properties like a standard ICA algorithm. Finally the guided GA (GGA) is modelled, at each step, as the stochastic matrix product acting on probability distributions over populations:

$$\mathbf{G^n} = \mathbf{P}_R^n \cdot \mathbf{F_n} \cdot \mathbf{C}_{\mathbf{P_c^n}}^{\mathbf{k}} \cdot \mathbf{M}_{(\mathbf{P_m},\mathbf{G})^{\mathbf{n}}} \qquad (27)$$

The GA used applies local search (using the selected mutation and crossover operators) around the values (or individuals) found to be optimal (elite) the last time. The computational time depends on the encoding length, number of individuals and genes. Because of the probabilistic nature of the GA-based method, the proposed method almost converges to a global optimal solution on average. In our simulation nonconvergent case was found.

Table 1. Figures: 1) Mean Crosstalk (50 runs) vs. iterations to reach the convergence for num. sources equal to 2 2) Mean Crosstalk (50 runs) vs. iterations to reach the convergence for num. sources equal to 20 3) Evolution of the crosstalk area vs. dimension. 4) Example of independent source used in the simulations.



7 Simulations

To check the performance of the proposed hybrid algorithm, 50 computer simulations were conducted to test the GGA vs. the GA method (Tan et al. 2001) and the most relevant ICA algorithm to date, FastICA (Hyvarinen et al. 1997). In this paper we described the evaluation of the computational complexity of the current methods in Table 2, also described in detail in several references such as (Górriz 2003) or (Tan et al. 2001). Somehow using a 8 nodes Cluster Pentium II 332MHz 512Kb Cache, the computational requirements of the algorithms (fitness functions, encoding, etc.) are generally negligible

Table 2. Complexity of the Hybrid Methods based on various approaches of the cost function described in section 5Note: N populations size, C Encoding size, ite iterations to convergence, $\tilde{\lambda}$ order of the approach used, n number of inputs.

`Complexity of generic cost function`	`mults`
	$(C-1)\left(\left(\tilde{\lambda}-n\right)\left(\tilde{\lambda}-n-1\right)+2\right)/2+$
	$+2^{\left(\tilde{\lambda}-n\right)\left(\tilde{\lambda}-n+1\right)/2}-1$
	`adds`
	$(C-1)\left(\left(\tilde{\lambda}-n\right)\left(\tilde{\lambda}-n-1\right)\right)/2+$
	$+C/2+2^{\left(\tilde{\lambda}-n\right)\left(\tilde{\lambda}-n+1\right)/2}$

`Complexity in function evaluation`	`mults` $N(Cn^2+n^2+C+2n+4)$ `adds` $N(n^2+2n+4C-5)$
`Number of evaluations`	$N \cdot ite$

compared with the cluster capacity. Logically GA-based BSS approaches suffer from a higher computational complexity as indicate in Table 2. Consider the mixing cases from 2 to 20 independent random super-gaussian input signals. We focuss our attention on the evolution of the crosstalk vs. the number of iterations using a mixing matrix randomly chosen in the interval $[-1,+1]$. The number of individuals chosen in the GA methods were $N_p = 30$ in the 50 (randomly mixing matrices) simulations for a number of input sources from 2 (standard BSS problem) to 20 (BSS in biomedicine or finances). The standard deviation of the parameters of the separation over the 50 runs never exceeded 1% of their mean values while using the FASTICA method we found large deviations from different mixing matrices due to its limited capacity of local search as dimension increases. The results for the crosstalk are displayed in Table 1. It can be seen from the simulation results that the FASTICA convergence rate decreases as dimension increases whereas GA approaches work efficiently. In practice we measure the independence of the outputs we express equation 13 in terms of higher order statistics (cumulants) using the characteristic function (or moment generating function) $\phi(\mathbf{k})$, where $\mathbf{k}$ is a vector of variables in the Fourier transform domain, and considering its natural logarithm $\Phi = log(\phi(\mathbf{k}))$. Thus we get:

$$Cum(\overbrace{y_i, y_j, \ldots}^{stimes}) = \kappa_s^i \delta_{i,j,\ldots} \qquad \forall i, j, \ldots \in [1, \ldots, n] \tag{28}$$

where $Cum(\overbrace{\ldots}^{stimes})$ is the s-th order cross-cumulant and $\kappa_s = Cum(\overbrace{y_i}^{stimes})$ is the auto-cumulant of order s straightforward related to moments (Chryssostomos et al. 1993). Hence vanishing cross-cumulants are a necessary condition for $y_1, \ldots, y_n$ to be independent[4]. Based on the above discussion, we can define the fitness function for BSS as:

[4] In practice we need independence between sources two against two.

$$f(p_o) = \sum_{i,j,\ldots} ||Cum(\overbrace{y_i, y_j, \ldots}^{stimes})|| \qquad \forall i,j,\ldots \in [1,\ldots,n] \tag{29}$$

where p_o is the parameter vector (individual) containing the separation matrix and $||\ldots||$ denotes the absolute value.

8 Conclusions

A GGA-based BSS method has been developed to solve BSS problem from the linear mixtures of independent sources. The proposed method obtain a good performance overcoming the local minima problem over multidimensional domains. Extensive simulation results prove the ability of the proposed method. This is particular useful in some medical applications where input space dimension increases and in real time applications where reaching fast convergence rates is the major objective.

References

Barlow, H.B, Possible principles underlying transformation of Sensory messages. Sensory Communication, W.A. Rosenblith, MIT Press, New York, U.S.A. (1961).

Bell,A.J., Sejnowski, T.J. An Information-Maximization Approach to Blind Separation and Blind Deconvolution. Neural Computation, vol 7, 1129-1159 (1995).

Hyvärinen, A., Oja, E., A fast fixed point algorithm for independent component analysis Neural Computation, 9: 1483-1492

Górriz, J.M., Algorítmos Híbridos para la Modelización de Series Temporales con Técnicas AR-ICA. PhD Thesis, University of Cádiz (2003)

Cao, X.R., Liu W.R., General Approach to Blind Source Separation. IEEE Transactions on signal Processing, vol 44, num 3, 562-571 (1996)

Michalewicz, Z., *Genetic Algorithms + Data structures = Evolution Programs*, springer Verlag, Berlin 1992.

Haggstrom, O., *Finite Markov Chains and Algorithmic Applications*, Cambridge University,1998.

Schmitt, L.M., Nehaniv, C.L., Fujii, R.H., *Linear Analysis of Genetic Algorithms*, Theoretical Computer Science, volume 200, pages 101-134, 1998.

Suzuki, J., *A markov Chain Analysis on Simple Genetic Algorithms*, IEEE Transaction on Systems, Man, and Cybernetics, vol 25, num=4, 655-659,(1995).

Chryssostomos, C., Petropulu, A.P., Higher Order Spectra Analysis: A Non-linear Signal Processing Framework Prentice Hall, London (1993)

Lozano, J.A., Larrañaga, P., Graña, M., Albizuri, F.X., *Genetic Algorithms: Bridging the Convergence Gap*, Theoretical Computer Science, vol 229, 11-22, (1999).

Rudolph, G., *Convergence Analysis of Canonical Genetic Algorithms*, IEEE Transactions on Neural Networks, vol 5, num 1,(1994) 96-101.

Tan, Y., Wang, J., Nonlinear Blind Source Separation Using Higher order Statistics and a Genetic Algorithm. IEEE Transactions on Evolutionary Computation, vol. 5, num 6 (2001)

Part X

Computer Security

RDWT Domain Watermarking based on Independent Component Analysis Extraction

[1]Thai Duy Hien, [1]Zensho Nakao, and [2]Yen-Wei Chen

[1]University of the Ryukyus, Okinawa 903-0213, Japan
{tdhien,nakao}@augusta.eee.u-ryukyu.ac.jp
[2]Ritsumeikan University, Shiga 525-8577, Japan
chen@is.ritsumei.ac.jp

Abstract. We present a new digital watermarking in which redundant wavelet transform (RDWT) is applied for watermark embedding and independent component analysis (ICA) is used to extract the watermark. By using RDWT, a large logo watermark can be embedded into transform coefficients. The advantage of using large logo is the ability to carry redundant information about copyright, increasing the robustness. In the embedding procedure, the watermark is embedded in RDWT domain. However, in the extraction procedure, the watermark is directly extracted from the watermarked image in spatial domain by an ICA-based detector. The experiment shows that the proposed scheme produces less image distortion than conventional DWT and is robust against Jpeg/Jpeg2000/SPIHT image compression and other attacks.

Keywords: Digital watermarking, redundant discrete wavelet transform, independent component analysis, intelligent watermarking.

1 Introduction

Digital watermarking has been introduced to provide copyright protection for digital data. Many designed watermarking for still image and video content have been proposed (Thai D. Hien *et al* 2004, Thai D. Hien *et al* 2003, L.Hua and J. E. Fowler 2002, M. Barni *et al* 1999, M. Barni *et al* 2001, S. Voloshynovskiy *et al* 2000), in which discrete wavelet transform (DWT) was used to embed spread spectrum watermark into DWT coefficients.

Recently, an alternative wavelet transform paradigm, RDWT, has been applied to digital watermarking schemes. L. Hua *et al* (2002) analyzed the performance of spread spectrum watermark when the transform is redundant transform rather than orthogonal expansion. RDWT is shift invariant, and its redundancy introduces an over complete frame expansion. It is known that frame expansion increases robustness to additive noise (M. J. Shensa 1992, V. K. Goyal *et al* 1998, Y. Xu *et al* 1994) which is the addition of noise to transform coefficients. Thus, RDWT based signal processing tends to be more robust than DWT based techniques.

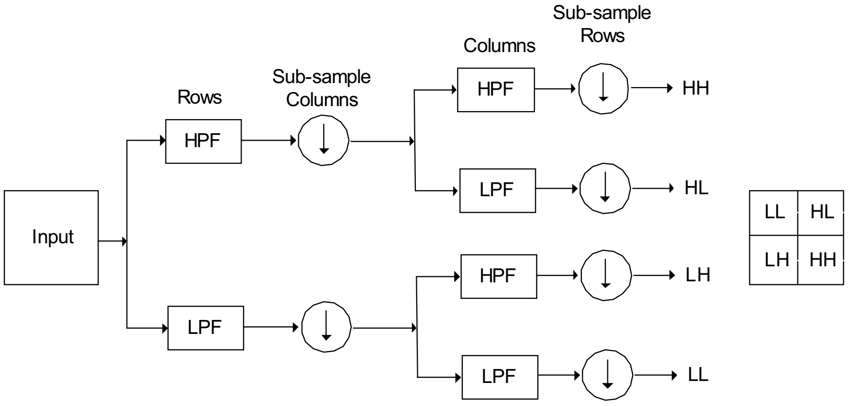
In this paper, a new RDWT based logo watermarking is proposed. Our scheme is adaptive because strength of the watermark is controlled by building a mask based on stochastic model. We take stochastic approach based on noise visibility function (NVF) (S. Voloshynovskiy *et al* 2000) for both robustness and adaptive watermarking algorithm. By knowing the stochastic model of the watermark and the host image, one can formulate weighting of the watermark information, and estimate capacity of the watermark. We use a large binary logo image of the same size as the original image. It is more robust against attacks because the extracted watermark which was degraded by attacks can still be verified with human eyes.

A new intelligent detection technique based on ICA is introduced for extraction to ensure the blind watermark. Furthermore, the watermark can be directly extracted from the watermarked image in spatial domain without going through any transform steps. Our experiment shows that RDWT-based digital watermarking produces less image distortion for the same watermark energy than conventional DWT and offers higher robustness against lossy image compression and various attacks.

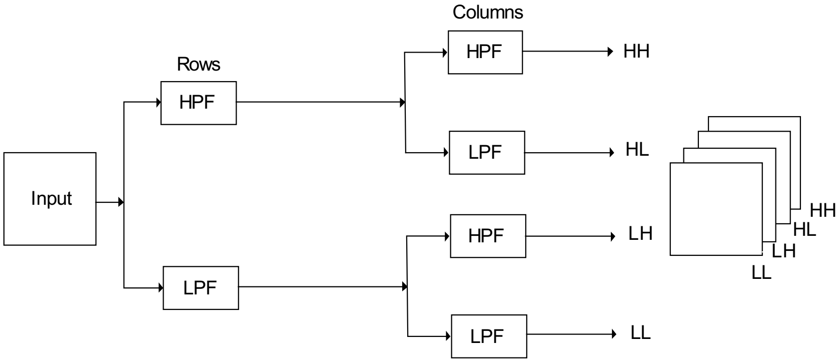
2 RWDT

The RDWT has been given several appellations over the years, including the “undecimated DWT,” the “over-complete DWT,” and the algorithme `a trous. The RDWT can be considered as an approximation to the continuous wavelet transform that removes the down-sampling operation from the traditional critically sampled DWT (Antonini *et al* 1992) to produce an over-complete representation. Redundant here means that the sub-sampling after each stage is omitted. The shift-variant characteristic of the DWT arises from its use of down-sampling, while the RDWT is shift invariant since the spatial sampling rate is fixed across scale.

RDWT has been proposed for signal detection and enhancement (M. J. Shensa 1992, V. K. Goyal et al 1998) since RDWT maintains uniform sampling rate in the time domain.



a. Non-Redundant Transform



b. Redundant Transform

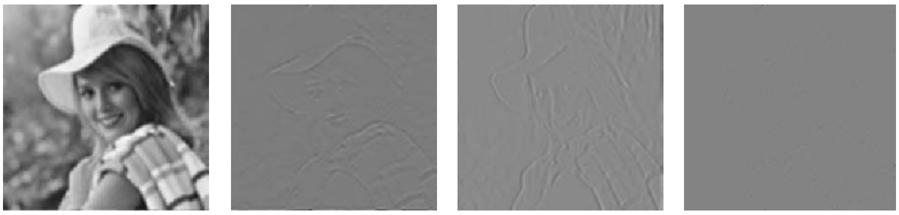
Fig.1. (a) Non-Redundant Transform (b) Redundant Transform

RDWT removes the decimation operators from DWT filter banks. To retain the multi-resolution characteristic, the wavelet filters must be adjusted accordingly at each scale. The lack of down-sampling in the RDWT analysis yields a redundant representation of the input sequence.

Fig.1 shows non-redundant transform and redundant transform; HPF and LPF are low-pass and high-pass filters, LL is approximation of the input image, HL is a detailed image containing information in the horizontal, LH is a detailed image containing information in the vertical, and HH is a detailed image containing information in diagonal. LL band at the highest level can be classified as most important, and the other 'detailed' bands can be classified as of lesser importance. The LL band can be decomposed once again in the same manner, thereby producing even more sub-bands. This can be done up to any levels.

The RDWT is a perfectly reconstructing transform. To invert the RDWT, one can simply independently invert each of the constituent critically sampled DWTs and average the resulting reconstructions together. However, this implementation of the inverse RDWT incurs unnecessary duplicate synthesis filtering of the high-pass bands; thus, one usually alternates between synthesis filtering and reconstruction averaging on a scale-by-scale basis.

Fig.2 shows redundant discrete wavelet transform of 512x512 “Elaine” at the second level: LL2 is the approximation of the input image, LH2 is the detailed image containing information in the vertical, HL2 is the detailed image containing information in the horizontal, and HH2 is the detailed image containing information in diagonal. LL2 band at the highest level can be classified as most important, and the other 'detailed' bands can be classified as less important. The LL2 band can be decomposed once again in the same manner, thereby producing even more sub-bands. This can be done up to any levels.

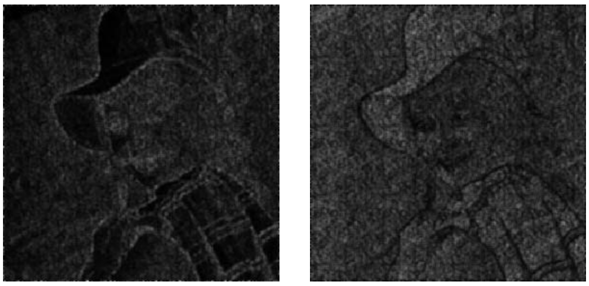


(a)LL2 (b)LH2 (c)HL2 (d)HH2

Fig.2. Redundant transform of 512x512 "Elaine" at the second level.

3 Adaptive Watermark Embedding

In this proposal, we use a binary logo watermark (-1, 1) of the same size as the host image. The pixels of the watermark are encrypted to form a new watermark image. Encryption is done by a linear feedback shift register "seed" of 128x128 blocks. We use the text binary image of size 512x512 as watermark. The encrypted watermark can be used for embedding, and de-encryption is required in the detection phase to recover the text image. The advantages of encryption of the watermark are: first, it secures the hidden message through encryption and, second, if repetitive patterns are left in place, it will remove those patterns which would affect the psycho-visual impact of embedding the watermark.



(a) (b)

Fig.3. A masking function at texture and edge region (a) and at flat region (b) of image "Elaine"

The watermark is embedded in RDWT coefficients. The image to be watermarked is decomposed into two levels by RDWT. Our embedding scheme is the same as (Thai D. Hien *et al* 2004), in which watermark is inserted in the mid frequency sub-bands by modifying the wavelet coefficients belonging to the two detailed sub-bands at the final level (LH_2 , HL_2). The stochastic model of cover image was applied to content adaptive watermark by computing NVF (S. Voloshynovskiy 2000). We consider non-stationary Gaussian model. The motivation of this model is its wide usage in image processing applications including image de-noising, restoration and compression. Figure 3 (a) shows masking function at texture and edge region and Fig. 3 (b) shows masking function at flat region of “Elaine” image.

The watermark which is encrypted as a pseudorandom binary $\{+1,-1\}$ sequence is inserted by modifying the wavelet coefficients belong to LH_2 , and HL_2 .

The watermark is embedded by the following equations:

$$\begin{aligned} LH_2^*(i,j) &= LH_2(i,j) + A^{LH}.\alpha(1-NVF(i,j))W(i,j) + B^{LH}.\beta.NVF(i,j)W(i,j) \\ HL_2^*(i,j) &= HL_2(i,j) + A^{HL}.\alpha(1-NVF(i,j))W(i,j) + B^{HL}.\beta.NVF(i,j)W(i,j) \end{aligned} \tag{1}$$

where LH_2^* , HL_2^* are watermarked transform coefficients, A^{LH} B^{LH} , A^{HL} B^{HL} denote the watermark strengths of texture/regions and flat region at I_l^{LH} I_l^{HL} sub-bands, respectively α , β are smoothing factors at the texture region and flat region. The watermark image $I^{'}$ is obtained by the inverse RDWT.

In order to apply ICA to blind watermark detection, the embedding process needs to create a private key for extraction. Private-key is created by linearly combining a public key k and the original media I. The public-key k is a binary image size 512x512 which has random value range (-1, 1): Private-key is then encrypted for security purposes.

Fig.4 shows the embedding scheme and the private-key K_p is created during the embedding phase.

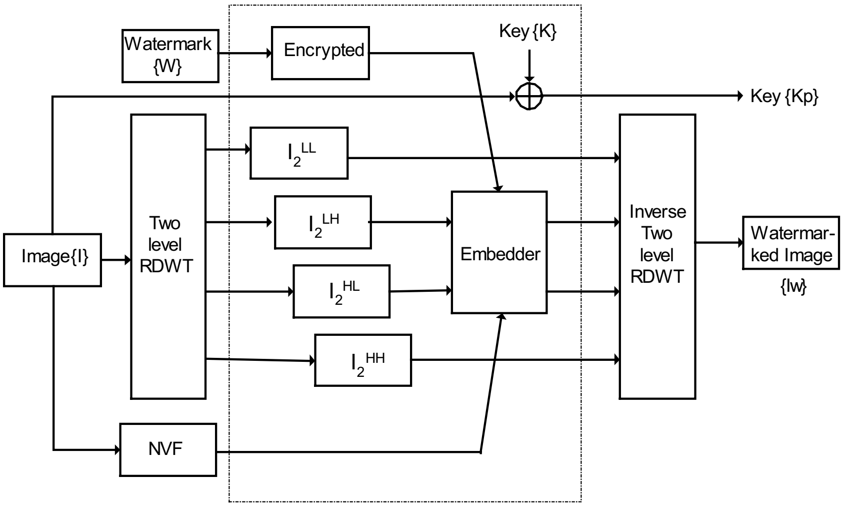


Fig.4. The proposed watermark embedding scheme

4 Intelligent Watermark Extraction Based on ICA

ICA is a method for extracting independent sources given only mixtures of the unknown sources (A. Hyvärinen and E. Oja 2000). It is a very general-purpose statistical technique to recover the independent sources given only sensor observation which are linear mixtures of independent source signals. The general model can be described as follows:

We start with the assumption that data vector $X = (x_1,..,x_M)$ can be represented in term of linear superposition of basis functions, $x = As = a_1 s_1 + ... + a_N s_N$, where $S = (s_1,..,s_N)$ is the coefficients, *A* is an *MxN* matrix and the columns $a_1,...a_N$ are called basis functions. Basis functions are consistent while coefficients vary with data. The goal of an efficient coding is to find a set of a_i which results in coefficient values being statistically as independent as possible over a set of data. Thus the linear analysis process is to find a matrix W such that $Y = Wx$, and to recover independent components *S* which were possibly permuted and rescaled. Rows of *W* correspond to columns of *A*.

The problem of watermark extraction can be considered as blind source separation (BSS) of mixtures. We introduced a method based on Independent Component Analysis (ICA) for extracting the watermark (Thai D. Hien *et al* 2004), which needs the transform process to get the LH and HL bands at the final levels.

In this paper, we introduce a new intelligent ICA detector which does not require the transform process to separate LH and HL bands for watermark extraction. ICA is applied directly on the watermarked image with the help of the private key. One can extract successfully the watermark to claim the ownership.

Mixtures are created by the following equations:

$$\begin{aligned} Mix_1 &= I' + K_p \\ Mix_2 &= K + K_p \\ Mix_3 &= I' \end{aligned} \tag{2}$$

where I' is the watermarked image, K is the public key and K_p is the private key. Those mixtures can be modeled as:

$$\begin{aligned} X_1 &= a_{11} I + a_{12} W + a_{13} k \\ X_2 &= a_{21} I + a_{22} W + a_{23} k \\ X_3 &= a_{31} I + a_{32} W + a_{33} k \end{aligned}$$

where *A* is the mixing matrix, *X* is the mixture, and *W* is the watermark. Applying the ICA algorithm for those mixtures, one can extract the watermark. Fig. 5 shows ICA blind watermarked extraction.

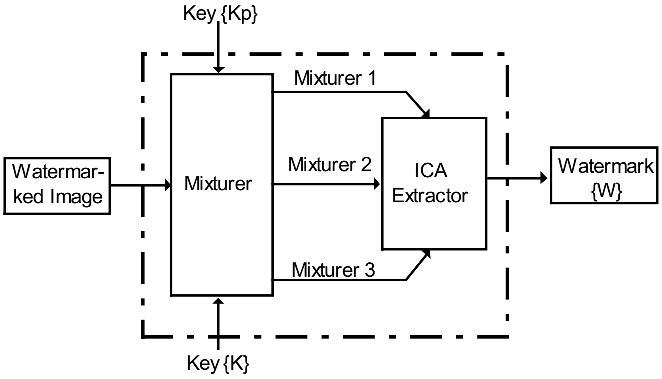


Fig.5. Proposed extraction.

After extracting the watermark, users can compare the extracted outcome with the referenced watermark subjectively. However, since factors such as image size, expertise of the observers and the experimental conditions affect the subjective measurement, the validation of the extracted out come must be quantitatively measured to objectively access the attracted fidelity. Therefore, a similarity measurement of the extracted, $W^{'}(i,j)$ and the referenced watermarks, $W(i,j)$ can be defined by the normalized correlation:

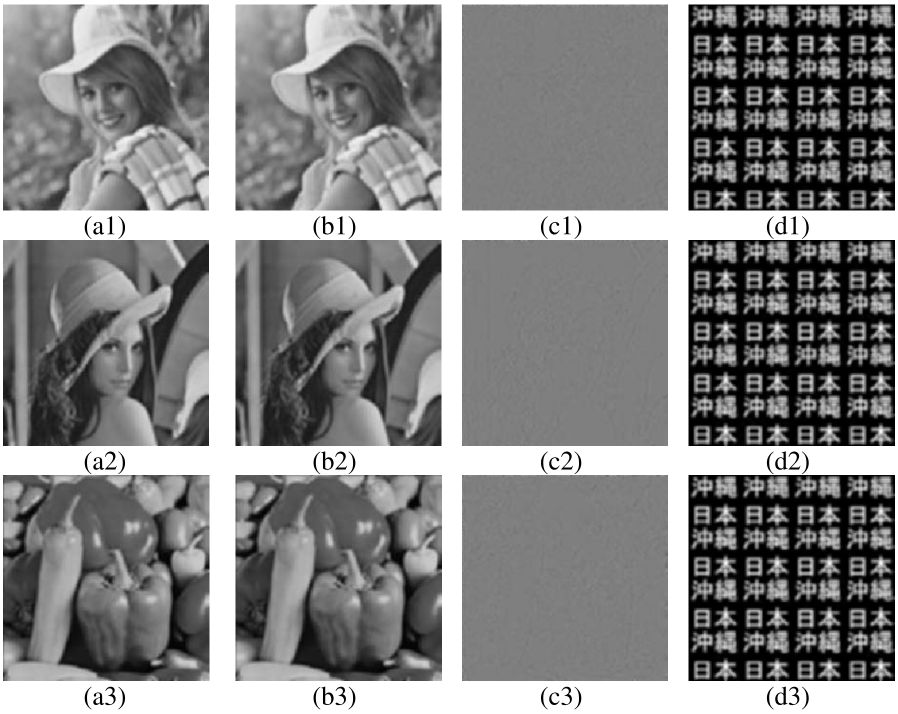
$$NC = \frac{\sum_{i=1}^{M_W}\sum_{j=1}^{N_W}\left[W(i,j)W^{'}(i,j)\right]}{\sum_{i=1}^{M_W}\sum_{j=1}^{N_W}\left[W(i,j)\right]^2} \qquad (3)$$

which is called cross-correlation normalized by refering watermark energy to give unity as the peak correlation, the value of $NC = [0,1]$, and if we acquire the higher NC values, the embedded watermark is more similar to the extracted one.

5 Experimental Results

The proposed method was tested on some standard images with size 512x512. Daubechies filters were used for computing RDWT with two levels. The Japanese text signature "Okinawa Japan" of size 512x512 is used as a watermark. The watermark is embedded in LH and HL bands at the second level RDWT.

There are several important requirements for effective watermarking: transparency, robustness and capacity. One of the important requirements of watermark is to compromise between the invisibility and the robustness. First, the watermark should be embedded in an invisible way to avoid degrading of perceptual quality. Second, the watermark should be robust against watermark attacks which are applied to image content. These attacks include but not limited to lossy image compression, filtering, noise-adding, and geometrical modification. The similarity measurement of the extracted watermarks $W^{'}(i,j)$ and the referenced watermark $W(i,j)$ can be defined by the normalized correlation (NC), and the value of NC lies in [0,1].



(a1) (b1) (c1) (d1)

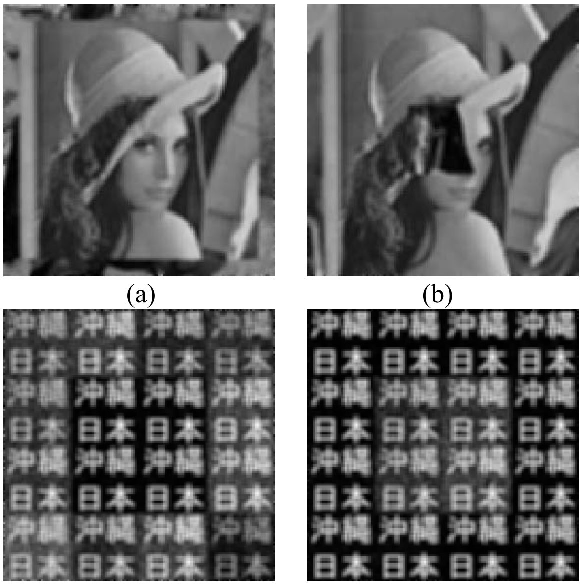
(a2) (b2) (c2) (d2)

(a3) (b3) (c3) (d3)

Fig.6. (a1) (a2) (a3) the “Elaine”, “Lena” and “Peppers” images, (b1) (b2) (b3) watermarked “Elaine”, “Lena” and “Peppers” images, (c1) (c2) (c3) difference between the original “Elaine”, “Lena” and “Peppers” and the watermarked ““Elaine”, “Lena” and “Peppers”, (d1) (d2) (d3) extracted watermarks from “Elaine”, “Lena” and “Peppers” by the ICA detector, respectively.

Experiments show that RDWT-based digital watermarking produces less image distortion for much watermark energy. Figure 6 (a1), (a2) and (a3) show the original image “Elaine”, “Lena” and “Peppers” with size 512x512, Fig.6 (b1), (b2) and (b3) are watermarked “Elaine”, “Lena” and “Peppers”. Both original and watermarked images are evidently indistinguishable, and the watermarked images have corresponding PNSR = 52.03dB, PNSR = 51.3dB and PNSR = 50.88dB. The effect of the masking function on the watermarked images is shown in Fig.6 (c1), (c2) and (c3) which are the difference between original images and the watermarked images, magnified by factor 8. Figure 6 (d1), (d2) and (d3) show extracted 512x512 watermarks of "Elaine", "Lena" and "Peppers" images with (NC = 1.00). The result confirms that our ICA detector can perfectly extract logo watermark and the algorithm is adaptive.

Experiments show that RDWT-based digital watermarking produces less image distortion for more watermark energy than conventional DWT. Therefore, the proposed RDWT embedding method will offer increased robustness against some attacks.



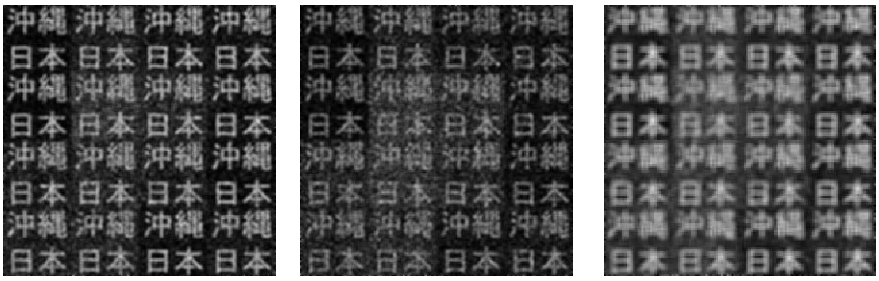
(a) (b)

(c) (d)

Fig.7. Cropped "Lena" image in experiments and corresponding extracted watermark.

Figure 7 shows the cropped "Lena" in experiments and its extracted results, 7(a) crop 45% of the surrounding and replace with "peppers" image, 7(b) center crop 12% and replace with "Man" image, 7(c) extracted output of 7(a), and 7(d) extracted output of 7(b).

Image compression is one of important attacks, the watermarked "Lena" image was compressed by Jpeg, Jpeg2000 and SPIHT, and results show that the watermark can still be extracted under very high compression. Figure 8(a) shows the ICA extraction from Jpeg2000 compression of watermarked image "Lena" with bit rate 0.4 bpp. Figure 8(b) shows ICA extraction from Jpeg compression with quality factor down to 10%. Figure 8(c) shows ICA extraction from SPIHT compression with bit rate 0.4 bpp. It shows that our proposal is very robust against lossy image compression.



(a) Jpeg 2000 compressed (rate = 0.4 bpp)

(b) Jpeg compressed (quality 10%)

(c) SPIHT compressed (rate = 0.4 bpp)

Fig.8. ICA extracted watermarks, (a) Jpeg 2000, (b) jpeg (c) SPIHT compression

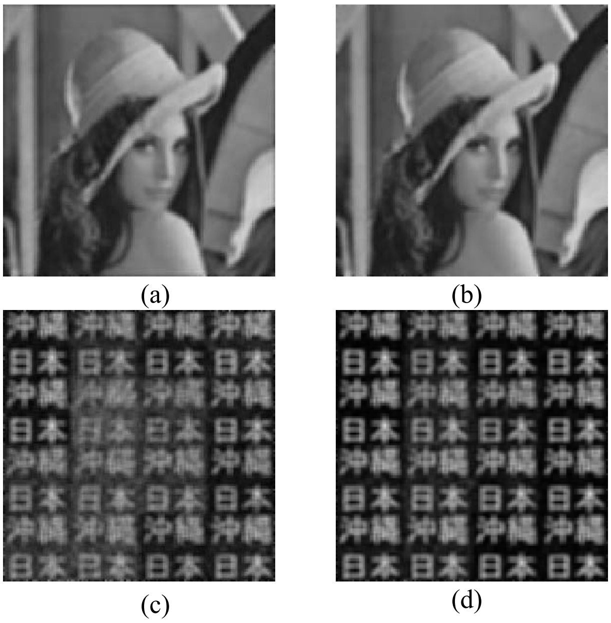


Fig.9. (a) Watermarked "Lena" image filtered with low-pass and (b) median filters 5x5 (c) and (d) their extracted watermark.

Figure 9 shows the watermarked "Lena" images filtered with low-pass and median filters, and their extracted watermarks. Table I shows PSNR/MSE and NC values from Jpeg compression of watermarked "Lena" image with quality factors from 85% to 15%.

JPEG2000 is intended to be a new and improved image compression method that replaces both JPEG and JBIG and it uses wavelet as the basis for both lossy and lossless algorithms. The watermarked image was coded and compressed under Jpeg2000. Table 2 shows PSNR/MSE and NC values from Jpeg2000 compression of watermarked “Lena” image with bit rates from 2.4 bits per pixel to 0.4 bits per pixel.

Table 1. PSNR/MSE and NC values of JPEG compression

Jpeg Quality Factors	PSNR/MSE	NC
15%	32.4353/ 37.1149	0.5770
25%	34.2840/ 24.2481	0.6270
45%	36.1640/ 15.7282	0.6698
65%	37.5554/ 11.4166	0.8162
85%	43.3334/ 6.2788	0.9979

Table 2. PSNR/MSE and NC values of Jpeg2000 compression

Bit rates	PSNR/MSE	NC
0.4	36.2844/ 15.2981	0.6859
0.8	39.3927/ 7.4784 1	0.7578
1.2	40.9852/ 5.1828	0.8342
1.6	43.3334 / 3.7537	0.8873
2.0	43.3334 / 3.0181	0.9155
2.4	44.3645/ 2.3803	0.9330

Table 3. PSNR/MSE and NC values of SPIHT compression

Bit rates	PSNR/MSE	NC
0.1	30.5149/ 57.7554	0.5322
0.2	33.5161/ 28.9380	0.6162
0.4	36.7877/ 13.6242	0.6639
0.8	40.0676/ 6.4020	0.7859
1.2	41.8551/ 4.2420	0.8781
1.6	43.3895/ 2.9794	0.9220
2.0	44.7878/ 2.1592	0.9486
2.4	46.3130/ 1.5198	0.9696

Image Compression with Set Partitioning in Hierarchical Trees (SPIHT) belongs to the next generation of wavelet encoders, employing more sophisticated coding. In fact, SPIHT exploits the properties of the wavelet-transformed images to increase its efficiency. The watermarked image was coded and compressed under SPIHT. Table 3 shows PSNR/MSE and NC values from SPIHT compression of watermarked "Lena" image with bit rates from 2.4 bits per pixel to 0.4 bits per pixel.

By the proposed RDWT technique, the watermark is distributed over the entire image with high energy, and image adaptive embedding is made. It is suitable for robustness against cropping attack. The use of large logo watermark will increase the robustness, and the proposed RDWT embedding method offers robustness against some attacks. Table 4 shows PSNR and NC values of the proposed method against some attacks.

Table 4. NC values vs. the attacking methods.

Attacks	PSNR(dB)	NC
Adding random noise (Power = 200)	25.13	0.90
Low-pass filtered (5x5)	29.14	0.64
Median filtered (5x5)	32.47	0.81
Crop 10% and replace with other image	21.14	0.91
Crop surrounding and replace with "peppers" image	17.93	0.72

6. Conclusions

This paper proposes an adaptive watermarking, which is based on RDWT for embedding a large logo watermark and a new intelligent ICA detector is introduced to detect the logo watermark in spatial domain. By RDWT we can embed a large logo watermark into a host image. The use of large logo watermark offers ability to carry redundant information about copyright and increasing the robustness. Experimental results demonstrated that the proposed embedding technique can survive under incoming Jpeg2000 compression and RDWT-based digital watermarking produces less image distortion for more watermark energy than conventional DWT. The system was also checked under various types of attacks and showed the robustness. The future work will evaluate our proposal under benchmarking program such as Checkmark and investigate the robustness under attacks.

Reference

A. Hyvärinen and E. Oja (2000), “Independent Component Analysis: Algorithms and Applications,” *Neural Networks*, vol. 13, Issue 4, pp. 411--430.

L. Hua and J. E. Fowler (2002), “A Performance Analysis of Spread-Spectrum Watermarking Based on Redundant Transforms,” *Proceedings of the IEEE International Conference on Multimedia and Expo*, Lausanne, Switzerland, vol. 2, pp. 553-556.

M. Barni, F. Bartolini, V. Cappellini, A. Lippi, A. Piva (1999), “DWT-based technique for spatio-frequency masking of digital signatures,” *Proceedings of SPIE* Vol. 3657, Security and Watermarking of Multimedia Contents.

M. Barni, F. Bartolini, and A. Piva (2001), "Improved Wavelet-Based Watermarking Through Pixel-Wise Masking," *IEEE Trans., Image Process*., vol.10, no. 5, pp. 783-791.

M. J. Shensa (1992), “The discrete wavelet transform: Wedding the `Atrous and mallat algorithms,” *IEEE Transactions on Signal Processing*, vol. 40, no. 10, pp. 2464–2482, October.

S. Voloshynovskiy, A. Herrigel, N. B., and T. Pun (2000), “A stochastic approach tocontent adaptive digital image watermarking,” *Lecture Notes in Computer Science*, vol.1768, pp. 212–236.

Thai D. Hien, Zensho Nakao, & Yen-Wei Chen (2004), “A Robust Logo Multiresolution Watermarking Based on Independent Component Analysis Extraction,” *Lecture Notes in Computer Science*, Vol.2939, Springer, pp 408-422.

Thai D. Hien, Yen-Wei Chen, & Zensho Nakao (2003), “PCA Based Digital Watermarking,” *Lecture Notes in Artificial Intelligence*, Vol. 2774, Springer, pp.1427-1434.

V. K. Goyal, M. Vetterli, and N. T. Thao(1998), “Quantized overcomplete expansions in IRn: Analysis, synthesis, and algorithms*,” IEEE Transactions on Information Theory*, vol. 44, no. 1, pp.16–31, January.

Y. Xu, J. B. Weaver, D. M. Healy, Jr., and J. Lu (1994), “Wavelet Transform Domain Filters: A Spatially Selective Noise Filtration Technique,” *IEEE Trans. Image Proc*., vol. 3, no. 6, pp.747-758, Nov. 1994.

Antonini, M.; Barlaud, M.; Mathieu, P.; Daubechies, I." Image coding using wavelet transform," *IEEE Transactions on Image Processing,* Vol. 1, No. 2, pp. 205-220, Apr 1992.

Towards Very Fast Modular Exponentiations Using Ant Colony

Nadia Nedjah[1] and Luiza de Macedo Mourelle[2]

[1] Department of Electronics Engineering and Telecommunication
Faculty of Engineering, State University of Rio de Janeiro,
Rua São Francisco Xavier, 524, Maracanã,
Rio de Janeiro, RJ, Brazil
nadia@eng.uerj.br
http://www.detel.eng.uerj.br

[2] Department of Systems Engineering and Computation
Faculty of Engineering, State University of Rio de Janeiro,
Rua São Francisco Xavier, 524, Maracanã,
Rio de Janeiro, RJ, Brazil
ldmm@eng.uerj.br
http://www.desc.eng.uerj.br

The performance of public-key cryptosystems is primarily determined by the implementation efficiency of the modular multiplication and exponentiation. As the operands, i.e. the plain text of a message or the cipher (possibly a partially ciphered) text are usually large (1024 bits or more), and in order to improve time requirements of the encryption/decryption operations, it is essential to attempt to minimise the number of modular multiplications performed. In this paper, we exploit the ant colony's principles to engineer a minimal addition chain that allows one to compute modular exponentiations very efficiently. The obtained results are compared to existing heuristics as well as to genetically evolved addition chains, i.e. using genetic algorithms.

1 Introduction

The modular exponentiation is a common operation for scrambling and is used by several public-key cryptosystems, such as the RSA encryption scheme [6]. It consists of a repetition of modular multiplications: $C = T^E \bmod M$, where T is the *plaintext* such that $0 \leq T < M$ and C is the *ciphertext* or vice-versa, E is either the *public* or the *private key* depending on whether T is the plaintext or the ciphertext, and M is called the *modulus*. The decryption and encryption operations are performed using the same procedure, i.e. using the modular exponentiation.

It is clear that one should not compute T^E then reduce the result *modulo* M as the space requirements to store T^E is $E \times \log_2 M$, which is huge.

A simple procedure to compute $C = T^E \bmod M$ based on the paper-and-pencil method requires $E - 1$ modular multiplications. It computes all powers of $T : T \rightarrow T^2 \rightarrow T^3 \rightarrow \ldots \rightarrow T^{E-1} \rightarrow T^E$.

The paper-and-pencil method computes more multiplications than necessary. For instance, to compute T^8, it needs 7 multiplications, i.e. $T \rightarrow T^2 \rightarrow T^3 \rightarrow T^4 \rightarrow T^5 \rightarrow T^6 \rightarrow T^7 \rightarrow T^8$. However, T^8 can be computed using only 3 multiplications $T \rightarrow T^2 \rightarrow T^4 \rightarrow T^8$. The basic question is: what is the fewest number of multiplications to compute T^E, given that the only operation allowed is multiplying two already computed powers of T? Answering this question is *NP*-complete, but there are several efficient algorithms that can find a near optimal ones.

Ant systems [1] are distributed multi-agent systems [3] that simulate real ant colony. Each agent behaves as an ant within its colony. Despite the fact that ants have very bad vision, they always are capable to find the shortest path from their nest to wherever the food is. To do so, ants deposit a trail of a chemical substance called *pheromone* on the path they use to reach the food. On intersection points, ants tend to choose a path with high amount of pheromone. Clearly, the ants that travel through the shorter path are capable to return quicker and so the pheromone deposited on that path increases relatively faster than that deposited on much longer alternative paths. Consequently, all the ants of the colony end using the shorter way.

In this paper, we exploit the ant strategy to obtain an optimal solution to addition chain minimisation *NP*-complete problem. In order to clearly report the research work performed, we subdivide the rest of this paper into five important sections. First, in Section 2, we describe the addition chain-based methods and state the minimisation problem. Then, in Section 3, we formally explain the principles of ant systems and algorithms. Thereafter, in Section 4, we provide and comment the data structures required for the bookkeeping of the reached (partial) addition chain as well as rules applied to represent and update the pheromone level. Moreover, we present the proposed multi-threaded implementation of the ant system. Subsequently, in Section 5, we expose the results obtained by the ant system and compare them to those evolved using genetic algorithms as well as to those exploited by traditional methods such as *m*-ary and sliding window methods. Finally, in Section 6, we conclude the paper and comment on the obtained results.

2 Addition Chain Minimisation

The addition chain-based methods use a sequence of positive integers such that the first number of the chain is 1 and the last is the exponent E, and in which each member is the sum of two previous members of the chain. For instance, the addition chain used by the paper-and-pencil method is

$(1, 2, 3, ..., E-2, E-1, E)$. A formal definition of an addition chain is given below:

Definition 1. An *addition chain* of length l for an positive integer n is a sequence of positive integers $(a_1, a_2, \ldots, a_l)$ such that $a_1 = 1, a_l = n$ and $a_k = a_i + a_j, 1 \leq i \leq j < k \leq l$.

2.1 Addition Chain-Based Methods

The algorithm used to compute the modular exponentiation $C = T^E \bmod M$ based on a given non-redundant addition chain, is specified in Algorithm 1, wherein PoT stands for the array of *P*owers *o*f T.

Algorithm 1. AdditionChainBasedMethod(T, M, E)

```
Let $(a_1 = 1, a_2, \ldots, a_l = E)$ be the addition chain;
$PoT[0] := T \bmod M$;
for $k := 1$ to $l$ do
    Let $a_k = a_i + a_j | i < k$ and $j < k$;
    $PoT[k] := PoT[i] \times PoT[j] \bmod M$;
return $PoT[l]$;
end.
```

Finding a minimal addition chain for a given number is *NP*-hard [2]. Therefore, heuristics were developed to attempt to approach such a chain. The most used heuristic consists of scanning the digits of E from the less significant to the most significant digit and grouping them in partitions P_i. The size of the partitions can be constant or variable [2], [4], [5]. Modular exponentiation methods based on constant-size partitioning of the exponent are usually called *m-ary*, where m is a power of two and $\log_2 m$ is the size of a partition while methods based on variable-size windows are usually called *sliding window*. There exist several strategies to partition the exponent. The generic computation on which these methods are based, is formalised in Algorithm 2, wherein $\Pi(E)$ consists of the set of partitions in exponent E and, V_i and L_i denote the value and length of partition P_i respectively.

Algorithm 2. HeuristicsBasedModularExpo(T, M, E)

```
Build $\Pi(E)$ using the given strategy;
for each $P_i$ in $\Pi(E)$ do Compute $T^{V_i} \bmod M$;
$C := T^{V_{b-1}} \bmod M$;
for $i := b-2$ downto $0$ do
    $C := T^{2^{L_i}} \bmod M$;
    if $V_i \neq 0$ then $C := C \times T^{V_i} \bmod M$;
return $C$;
end.
```

2.2 Addition Chain Minimisation Problem

It is perfectly clear that the shorter the addition chain is, the faster Algorithm 1. Consequently, the addition chain minimisation problem consists of finding a sequence of numbers that constitutes an addition chain for a given exponent. The sequence of numbers should be of a minimal length.

The heuristics used the m-ary and sliding window methods, described in Algorithm 2, generate a relatively short addition chains. However, the performance of modular exponentiation can be further improved if the addition chain is much shorter.

In previous research work, we applied genetic algorithms to evolve a minimal addition chains. Indeed, the application of genetic algorithms produced much shorter addition chains, compared with those of the m-ary and sliding window methods. Interested readers can find further details in [4].

In this paper, we describe an ant system that applies ant colony principles to the addition chain minimisation problem. We show that this ant system finds shorter addition chains, compared with the ones based on the heuristics of Algorithm 2 and to those evolved by the genetic algorithms.

3 Ant Systems and Algorithms

Ant systems can be viewed as multi-agent systems [1] that use a shared memory through which they communicate and a local memory to store the locally reached problem solution. Fig. 1. depicts the overall structure of an system, wherein A_i and LM_i represent the $i^{th.}$ agent of the ant system and its local memory respectively. Mainly, the shared memory (SM) holds the pheromone information while the local memory LM_i keeps the solution (possibly partial) that agent A_i reached so far.

The behaviour of an artificial ant colony is summarised in Algorithm 3, wherein N, C, SM are the number of artificial ants that form the colony, the characteristics of the expected solution and the shared memory used by the artificial ants to store pheromone information respectively.

In Algorithm 3, the first step consists of activating N distinct artificial ants that should work simultaneously. Every time an ant concludes its search, the shared memory is updated with an amount of pheromone, which should be proportional to the quality of the reached solution. This is called *global* pheromone update. When the solution yield by an ant's work is suitable (i.e. fits characteristic C) then all the active ants are stopped. Otherwise, the process is iterated until an adequate solution is encountered.

The behaviour of an artificial ant is described in Algorithm 4, wherein A_i and LM_i represent the ant identifier and the ant local memory, in which it stores the solution computed so far. First, the ant computes the probabilities that it uses to select the next state to move to. The computation depends on the solution built so far, the problem constraints as well as some heuristics

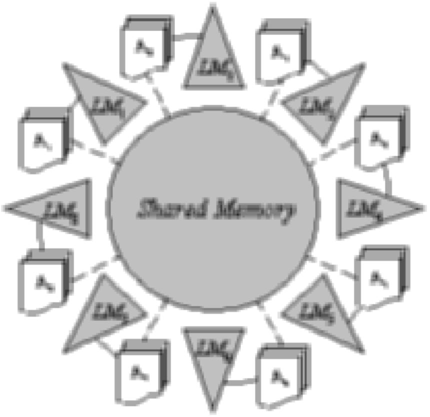


Fig. 1. Multi-agent system architecture

[7], [1]. Thereafter, the ant updates the solution stored in its local memory, deposits some *local* pheromone into the shared memory then moves to the chosen state. This process is iterated until complete problem solution is yielded.

Algorithm 3. ArtificialAntColony(N, C)

```
Initialise $SM$ with initial pheromone;
while Characteristics$(S) \neq C$ do
    for $i := 1$ to $N$ do
        Start ArtificialAnt$(A_i, LM_i)$; $Active := Active \cup \{A_i\}$;
    while (Characteristics$(S) \neq C$) or ($Active \neq \emptyset$) do
        Update $SM$ w.r.t. pheromone evaporation periodically;
        when an ant (say $A_i$) halts do
            $Active := Active \setminus \{A_i\}$; $\Phi :=$ Pheromone$(LM_i)$
            Update $SM$ with global pheromone $\Psi$;
            $S :=$ ExtractSolution$(LM_i)$;
        Stop ant $A_i \mid A_i \in Active$; $Active := Active \setminus \{A_i\}$;
return $S$;
end.
```

Algorithm 4. ArtificialAnt(A_i, LM_i)

```
Initialise $LM_i$;
while $CurrentState \neq TargetState$ do
    $P :=$ TransitionProbabilities$(LM_i)$;
    $NextState :=$ StateDecision$(LM_i, P)$; Update $LM_i$;
    Update $SM$ with local pheromone; $CurrentState := NextState$;
Stop $A_i$;
end.
```

4 Addition Chain Minimisation Using Ant System

In this section, we concentrate on the specialisation of the ant system of Algorithm 3 and Algorithm 4 to the addition chain minimisation problem. For this purpose, we describe how the shared and local memories are represented. We then detail the function that yields the solution (possibly partial) characteristics. Thereafter, we define the amount of pheromone to be deposited with respect to the solution obtained so far. Finally, we show how to compute the necessary probabilities and make the adequate decision towards a good addition chain for the considered exponent E.

4.1 The Ant System Shared Memory

The ant system shared memory is a two-dimension triangular array. The array has E rows. The number of columns depends on the row. It can be computed as in (1), wherein NC_i denotes the number of columns in row i.

$$NC_i = \begin{cases} 2^{i-1} - i + 1 & \text{if } 2^{i-1} \leq E \\ \\ E - i + 1 & \text{otherwise} \end{cases} \tag{1}$$

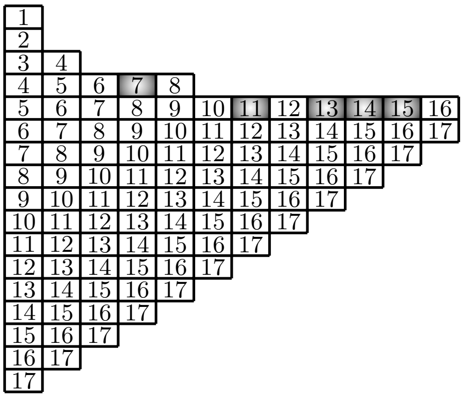
An entry $SM_{i,j}$ of the shared memory holds the pheromone deposited by ants that used exponent $i+j$ as the i th. member in the built addition chain. Note that $1 \leq i \leq E$ and for row i, $0 \leq j \leq NC_i$. Fig. 2 gives an example of the shared memory for exponent 17. In this example, a table entry is set to show the exponent corresponding to it. The exponent $E_{i,j}$ corresponding to entry $SM_{i,j}$ should be obtainable from exponents from previous rows. (2) formalises such a requirement.

$$E_{i,j} = E_{k_1,l_1} + E_{k_2,l_2} | \;\; 1 \leq k_1, k_2 < i, 0 \leq l_1, l_2 \leq j, k_1 = k_2 \Longleftrightarrow l_1 = l_2 \tag{2}$$

Note that, in Fig. 2, the exponents in the shaded entries are not valid exponents as for instance exponent 7 of row 4 is not obtainable from the sum of two previous different stages, as described in (2). The computational process that allows us to avoid these exponents is of very high cost. In order to avoid using these few exponents, we will penalise those ants that use them and hopefully, the solutions built by the ants will be almost all valid addition chains.

4.2 The Ant Local Memory

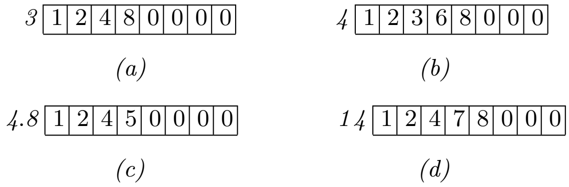
Each ant is endowed a local memory that allows it to store the solution or the part of it that was built so far. This local memory is divided into two parts: the first part represents the (partial) addition chain found by the ant so far and consists of a one-dimension array of E entries; the second part holds the *characteristics* of the solution. The characteristic of a solution represents its fitness to the objective of the optimisation. The details of how to compute a possibly partial addition chain



1											
2											
3	4										
4	5	6	7	8							
5	6	7	8	9	10	11	12	13	14	15	16
6	7	8	9	10	11	12	13	14	15	16	17
7	8	9	10	11	12	13	14	15	16	17	
8	9	10	11	12	13	14	15	16	17		
9	10	11	12	13	14	15	16	17			
10	11	12	13	14	15	16	17				
11	12	13	14	15	16	17					
12	13	14	15	16	17						
13	14	15	16	17							
14	15	16	17								
15	16	17									
16	17										
17											

Fig. 2. Example of the shared memory content for $E = 17$

are given in the next section. Fig. 3 shows four different examples of an ant local memory for exponent 8. Fig. 3(a) represents addition chain (1, 2, 4, 8), which is a valid and complete solution of fitness 3 while Fig. 3(b) depicts addition chain (1, 2, 4, 6, 8), which is also a valid and complete solution but of fitness 4. Fig. 3(c) represents partial addition chain (1, 2, 4, 5), which is a valid and but incomplete solution for exponent 8 and Fig. 3(d) consists of non-valid addition chain (1, 2, 4, 7, 8) as 7 is not the sum of any of the previous exponents.



3 | 1 | 2 | 4 | 8 | 0 | 0 | 0 | 0 |

(a)

4 | 1 | 2 | 3 | 6 | 8 | 0 | 0 | 0 |

(b)

4.8 | 1 | 2 | 4 | 5 | 0 | 0 | 0 | 0 |

(c)

14 | 1 | 2 | 4 | 7 | 8 | 0 | 0 | 0 |

(d)

Fig. 3. Example of an ant local memory: *(a)* complete valid solution of fitness 3; *(b)* complete valid solution of fitness 4; *(c)* partial valid solution of fitness 4.8; *(d)* complete non valid solution of fitness 14

4.3 Addition Chain Characteristics

The fitness evaluation of addition chain is done with respect to two aspects: *(i)* how much a given addition chain adheres to the Definition 1, i.e. how many members of the addition chain cannot be obtained summing up two previous members of the chain; *(ii)* how far the addition chain is reduced, i.e. what is the length of the addition chain. Equation (3) shows how to compute the fitness of an addition chain.

For a valid complete addition chain, the fitness coincides with its length, which is the number of multiplications that are required to compute the exponentiation using the addition chain. For a valid but incomplete addition chain, the fitness is a the *relative* length of the chain. It takes into account the distance between exponent E and the last exponent in the partial addition chain. Note that valid incomplete addition chains may have the same fitness of some other valid and complete addition chain. For instance, addition chains (1, 2, 3, 6, 8) and (1, 2, 3, 6) for exponent 8 have the same fitness 4.

For an invalid addition chain, a penalty, larger than E, is introduced into the fitness value for each exponent for which one cannot find two (may be equal) members of the chain whose sum is equal to the exponent in question. In the chains of Fig. 3, the penalty is 10.

$$Fitness(E, (a_1, a_2, \ldots, a_n)) = \frac{E \times (n-1)}{a_n} + \sum_{\substack{k \mid \ 3 \leq k \leq n \\ \forall i,j, \ 1 \leq i,j < k, \\ a_k \neq a_i + a_j}} penalty \qquad (3)$$

4.4 Pheromone Trail and State Transition Function

There are three situations wherein the pheromone trail is updated: *(a)* when an ant chooses to use exponent $F = i + j$ as the *i*th. member in its solution, the shared memory cell $SM_{i,j}$ is incremented with a constant value of pheromone; *(b)* when an ant halts because it reached a complete solution, say $A = (a_1, a_2, \ldots, a_n)$, all the shared memory cells $SM_{i,j}$ such that $i + j = a_i$ are incremented with pheromone value of $1/Fitness(A)$. Note that the better is the reached solution, the higher is the amount of pheromone deposited in the shared memory cells that correspond to the addition chain members. *(c)* The pheromone deposited should evaporate. Periodically, the pheromone amount stored in $SM_{i,j}$ is decremented in an exponential manner [7].

An ant, say A that has constructed partial addition chain $(a_1, a_2, \ldots, a_i, 0, \ldots, 0)$ for exponent E, is said to be in *step i.* In step $i + 1$, it may choose exponent a_{i+1} as one of exponents $a_i + 1, a_i + 2, \ldots, 2a_i$, if $2a_i \leq E$. That is, the ant may choose one of the exponents that are associated with the shared memory cells $SM_{i+1,a_i-i}, SM_{i+1,a_i-i+1}, \ldots, SM_{i+1,2a_i-i-1}$. Otherwise (i.e. if $2a_i > E$), it may only select from exponents $a_i + 1, a_i + 2, \ldots, E$. In this case, ant A may choose one of the exponent associated with $SM_{i+1,a_i-i}, SM_{i+1,a_i-i+1}, \ldots, SM_{i+1,E-i-1}$. Furthermore, ant A chooses the new exponent a_{i+1} with the probability of $P_{i,j}$ of (4).

$$P_{i,j} = \begin{cases} \frac{SM_{i+1,j}}{\max_{k=a_i-i}^{2a_i-i-1} SM_{i+1,k}} & \text{if } 2a_i \leq E \ \& \ j \in [a_i - i, 2a_i - i - 1] \\ \\ \frac{SM_{i+1,j}}{\max_{k=a_i-i}^{E-i-1} SM_{i+1,k}} & \text{if } 2a_i > E \ \& \ j \in [a_i - i, E - i - 1] \quad) \\ \\ 0 & \text{otherwise} \end{cases} \qquad (4)$$

5 Performance Comparison

The ant system described in Algorithm 3 was implemented using Java as multi-threaded ant system. Each ant was simulated by a thread that implements the artificial ant computation of Algorithm 4. A Pentium IV-HT$^{\text{TM}}$ of 2GB was used to run the ant system and obtain the performance results.

We compared the performance of the m-ary methods to the genetic algorithm and ant system-based methods. The average lengths of the addition chains for different exponents obtained by using these methods are given in Table 1. The exponent size is that of its binary representation (i.e. number of bits). The ant system-based method always outperforms all the others, including the genetic algorithm-based method [5]. The chart Fig. 4 shows the relation between the average length of the used addition chains.

Table 1. Average length of addition chain for binary, quaternary and octal method vs. genetic algorithm and ant system-based methods

exponet size	Binary	Quaternary	Octal	Genetic Algorithms	Ant System
32	47	43	43	41	38
64	95	87	85	79	71
128	191	175	167	158	145
256	383	351	333	320	277
512	767	703	663	630	569
1024	1535	1407	1325	1288	1022

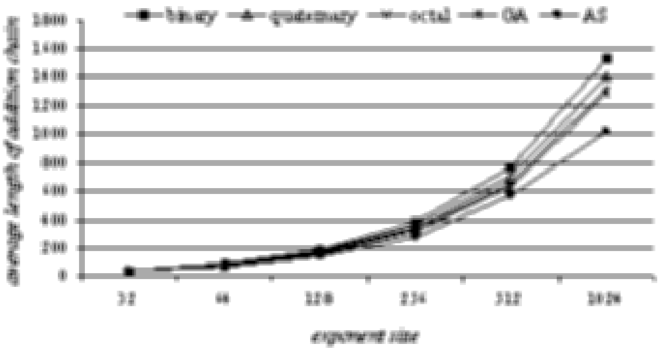


Fig. 4. Comparison of the average length of the addition chains for the binary, quaternary and octal methods vs. genetic algorithms and ant system-based methods

6 Conclusion

In this paper we applied the methodology of ant colony to the addition chain minimisation problem. We described how the shared and local memories are represented. We detailed the function that computes the solution fitness. We defined the amount of pheromone to be deposited with respect to the solution obtained by an ant. We showed how to compute the necessary probabilities and make the adequate decision towards a good addition chain for the considered exponent. We implemented the ant system described using multi-threading (each ant of the system was implemented by a thread). We compared the results obtained by the ant system to those of m-ary methods (binary, quaternary and octal methods). We also compared the obtained results to those obtained by the genetic algorithm. The ant system always finds a shorter addition chain.

References

1. Dorigo, M. and Gambardella, L.M. (1997), Ant Colony: a Cooperative Learning Approach to the Travelling Salesman Problem, IEEE Transaction on Evolutionary Computation, Vol. 1, No. 1, pp. 53-66
2. Downing, P. Leong B. and Sthi, R. (1981), Computing Sequences with Addition Chains, SIAM Journal on Computing, vol. 10, No. 3, pp. 638-646
3. Feber, J. (1995), Multi-Agent Systems: an Introduction to Distributed Artificial Intelligence, Addison-Wesley
4. Nedjah, N., Mourelle, L.M. (2002), Efficient Parallel Modular Exponentiation Algorithm, Second International Conference on Information systems, ADVIS'2002, Lecture Notes in Computer Science, Springer-Verlag, vol. 2457, pp. 405-414
5. Nedjah, N. and Mourelle, L.M., Minimal addition chains using genetic algorithms, Proceedings of the Fifteenth International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems, Lecture Notes in Computer Science, Springer-Verlag, vol. 2358, pp. 88-98
6. Rivest, R., Shamir, A. and Adleman, L. (1978), A method for Obtaining Digital Signature and Public-Key Cryptosystems, Communications of the ACM, 21:120-126
7. Stutzle, T. and Dorigo, M. (1999), ACO Algorithms for the Travelling Salesman Problems, Evolutionary Algorithms in Engineering and Computer Science, John-Wiley

Taming the Curse of Dimensionality in Kernels and Novelty Detection

Paul F. Evangelista
Department of Systems Engineering
United States Military Academy
West Point, NY 10996

Mark J. Embrechts
Department of Decision Sciences and Engineering Systems
Rensselaer Polytechnic Institute
Troy, New York 12180

Boleslaw K. Szymanski
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, New York 12180

Summary. The curse of dimensionality is a well known but not entirely well-understood phenomena. Too much data, in terms of the number of input variables, is not always a good thing. This is especially true when the problem involves unsupervised learning or supervised learning with unbalanced data (many negative observations but minimal positive observations). This paper addresses two issues involving high dimensional data: The first issue explores the behavior of kernels in high dimensional data. It is shown that variance, especially when contributed by meaningless noisy variables, confounds learning methods. The second part of this paper illustrates methods to overcome dimensionality problems with unsupervised learning utilizing subspace models. The modeling approach involves novelty detection with the one-class SVM.

1 Introduction

High dimensional data often create problems. This problem is exacerbated if the training data is only one class, unknown classes, or significantly unbalanced classes. Consider a binary classification problem that involves computer intrusion detection. Our intention is to classify network traffic, and we are interested in classifying the traffic as either attacks (intruders) or non attacks. Capturing network traffic is simple - hookup to a LAN cable, run tcpdump, and you can fill a hard drive within minutes. These captured network connections can be described with attributes; it is not uncommon for a network connection to be described with over 100 attributes [14]. However, the class of each connection will be unknown, or perhaps with reasonable confidence we can assume that all of the connections do not involve any attacks.

The above scenario can be generalized to other security problems as well. Given a matrix of data, $\mathbf{X}$, containing N observations and m attributes, we are interested in classifying this data as either potential attackers (positive class) or non attackers (negative class). If m is large, and our labels, $\mathbf{y} \in \mathbb{R}^{N \times 1}$, are unbalanced (usually plenty of known non attackers and few instances of attacks), one class (all non attackers), or unknown, increased dimensionality rapidly becomes a problem and feature selection is not feasible due to the minimal examples (if any) of the attacker class.

2 Recent Work

The primary model explored will be the one-class SVM. This is a novelty detection algorithm originally proposed in [27]. The model is relatively simple but a powerful method to detect novel events that occur after learning from a training set of normal events. Formally stated, the one-class SVM considers $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N \in \mathcal{X}$ instances of training observations and utilizes the popular "kernel trick" to introduce a non linear mapping of $\mathbf{x}_i \rightarrow \Phi(\mathbf{x}_i)$. Under Mercer's theorem, it is possible to evaluate the inner product of two feature mappings, such as $\Phi(\mathbf{x}_i)$ and $\Phi(\mathbf{x}_j)$, without knowing the actually feature mapping. This is possible because $\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \equiv \kappa(\mathbf{x}_i, \mathbf{x}_j)$ [2]. Φ will be considered a mapping into the feature space, F, from $\mathcal{X}$.

The following minimization function attempts to squeeze R, which can be thought of as the radius of a hypersphere, as small as possible in order to fit all of the training samples. If a training sample will not fit, ζ_i is a slack variable to allow for this. A free parameter, $\nu \in (0, 1)$, enables the modeler to adjust the impact of the slack variables.

$$\min_{R \in \mathbb{R}, \zeta \in \mathbb{R}^N, c \in F} \quad R^2 + \frac{1}{\nu N} \sum_i \zeta_i \tag{1}$$

$$\text{subject to} \quad \| \Phi(\mathbf{x}_i) - c \|^2 \leq R^2 + \zeta_i, \quad \zeta_i \geq 0 \text{ for } i \in [N]$$

The lagrangian dual of the one class SVM is shown below in equation 2.

$$\max_\alpha \sum_i \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j} \alpha_i \alpha_j \kappa(\mathbf{x}_i, \mathbf{x}_j) \tag{2}$$

$$\text{subject to} \quad 0 \leq \alpha_i \leq \frac{1}{\upsilon N} \text{ and } \sum_i \alpha_i = 1$$

Cristianini and Shawe-Taylor provide a detailed explanation of one-class SVMs in [24]. Stolfo and Wang [25] successfully apply the one-class SVM to the SEA dataset and compare it with several of the techniques mentioned above. Chen uses the one-class SVM for image retrieval [8]. Schölkopf et. al. explore the above formulation of the one-class SVM and other formulations in [23]. Fortunately there is also freely available software that implements the one-class SVM, written in C++ by Chang and Lin [7].

The dimensionality problem faced by the one-class SVM has been mentioned in several papers, however it is typically a "future works" type of discussion. Tax and Duin clearly mention that dimensionality is a problem in [27], however they offer no suggestions to overcome this. Modeling in subspaces, which is the proposed method to overcome this problem, is not an altogether novel concept. In data mining, subspace modeling to overcome dimensionality is a popular approach. Aggarwal discusses this in [1]. Parsons et. al. provide a survey of subspace clustering techniques in [21]. The curse of dimensionality is largely a function of class imbalance and our apriori knowledge of the distribution of $(\mathbf{x}|\mathbf{y})$. This implies that the curse of dimensionality is a problem that impacts unsupervised problems the most severely, and it is not surprising that data mining clustering algorithms, an unsupervised method, has come to realize the value of modeling in subspaces.

3 Analytical Investigation

3.1 The Curse of Dimensionality, Kernels, and Class Imbalance

Machine learning and data mining problems typically seek to show a degree of similarity between observations, often as a distance metric. Beyer et. al. discuss the problem of high dimensional data and distance metrics in [3], presenting a probabilistic approach and illustrating that the maximally distant point and minimally distant point converge in distance as dimensionality increases. A problem with distance metrics in high dimensional space is that distance is typically measured across volume. Volume increases exponentially as dimensionality increases, and points tend to become equidistant. The curse of dimensionality is explained with several artificial data problems in [15].

Kernel based pattern recognition, especially in the unsupervised domain, is not entirely robust against high dimensional input spaces. A kernel is nothing more than a similarity measure between two observations. Given two observations, $\mathbf{x_1}$ and $\mathbf{x_2}$, the kernel between these two points is represented as $\kappa(\mathbf{x_1}, \mathbf{x_2})$. A large value for $\kappa(\mathbf{x_1}, \mathbf{x_2})$ indicates similar points, where smaller values indicate dissimilar points. Typical kernels include the linear kernel, $\kappa(\mathbf{x_1}, \mathbf{x_2}) = \langle \mathbf{x_1}, \mathbf{x_2} \rangle$, the polynomial kernel, $\kappa(\mathbf{x_1}, \mathbf{x_2}) = (\langle \mathbf{x_1}, \mathbf{x_2} \rangle + 1)^p$, and the popular gaussian kernel, $\kappa(\mathbf{x_1}, \mathbf{x_2}) = e^{(-\|\mathbf{x_1}-\mathbf{x_2}\|/2\sigma^2)}$. As shown, these kernels are all functions of inner products. If the variables within $\mathbf{x_1}$ and $\mathbf{x_2}$ are considered random variables, these kernels can be modeled as functions of random variables. The fundamental premise of pattern recognition is the following:

$$(\kappa(\mathbf{x_1}, \mathbf{x_2})|y_1 = y_2) > (\kappa(\mathbf{x_1}, \mathbf{x_2})|y_1 \neq y_2) \tag{3}$$

If this premise is consistently true, good performance occurs. By modeling these kernels as functions of random variables, it can be shown that the addition of noisy, meaningless input variables degrades performance and the likelihood of the fundamental premise shown above.

In a classification problem, the curse of dimensionality is a function of the degree of imbalance. If there are a small number of positive examples to learn from, feature selection is possible but difficult. With unbalanced data, significant evidence is required to illustrate that a feature is not meaningful. If the problem is balanced, the burden is not as great. Features are much more easily filtered and selected.

A simple explanation of this is to consider a two sample Kolmogorov test [22]. This is a classical statistical test to determine whether or not two samples come from the same distribution, and this test is general regardless of the distribution. In classification models, a meaningful variable should behave differently depending on the class, implying distributions that are not equal. Stated in terms of distributions, if x is any variable taken from the space of all variables in the dataset, $(F_x(x)|y = 1)$ should not be equivalent to $(G_x(x)|y = -1)$. $F_x(x)$ and $G_x(x)$ simply represent the cumulative distribution functions of $(x|y = 1)$ and $(x|y = -1)$, respectively. In order to apply the two sample Kolmogorov test, the empirical distribution functions of $F_x(x)$ and $G_x(x)$ must be calculated from a given sample, and these distribution functions will be denoted as $F^*_{N_1}(x)$ and $G^*_{N_2}(x)$. N_1 will equate to the number of samples in the minority class, and N_2 equates to the number of samples in the majority class. These empirical distribution functions are easily derived from the order statistics of the given sample, which is shown in [22]. The Kolmogorov two sample test states that if the supremum of the difference of these functions exceeds a tabled critical value depending on the modeler's choice of α(sum of probabilities in two tails), then these two distributions are significantly different. Stated formally, our hypothesis is that $F_x(x) = G_x(x)$. We reject this hypothesis with a confidence of $(1 - \alpha)$ if equation 4 is true.

$$D_{N_1,N_2} = \sup_{-\infty<x<\infty} |F^*_{N_1}(x) - G^*_{N_2}(x)| > D_{N_1,N_2,\alpha} \tag{4}$$

For larger values of N_1 and N_2 (both N_1 and N_2 greater than 20) and $\alpha = .05$, we can consider equation 5 to illustrate an example. This equation is found in the tables listed in [22]:

$$D_{N_1,N_2,\alpha=.05} = 1.36\sqrt{\frac{N_1 + N_2}{N_1 N_2}} \tag{5}$$

If N_2 is fixed at 100, and N_1 is considered the minority class, it is possible to plot the relationship between m and the critical value necessary to reject the hypothesis.

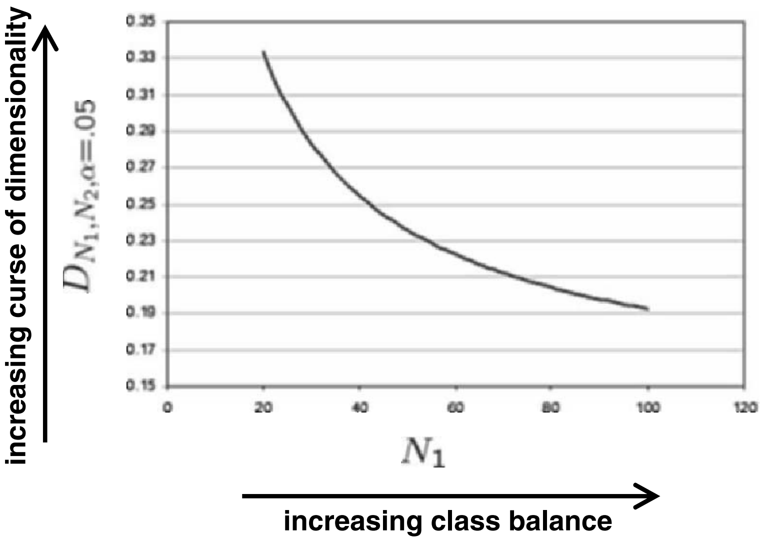


Fig. 1. Plot of critical value for two sample Kolmogorov test with fixed N_2, $\alpha = .05$

Figure 1 illustrates the effect of class imbalance on feature selection. If the classes are not balanced, as is the case when $N_1 = 20$ and $N_2 = 100$, there is a large value required for D_{N_1,N_2}. It is also evident that if the classes were more severely imbalanced, D_{N_1,N_2} would continue to grow exponentially. As the classes balance, D_{N_1,N_2} and the critical value begins to approach a limit. The point of this exercise was to show that the curse of dimensionality is a function of the level of imbalance between the classes, and the two sample Kolmogorov test provides a compact and statistically grounded explanation for this.

3.2 Kernel Behavior in High Dimensional Input Space

An example is given in this section which illustrates the impact of dimensionality on linear kernels and gaussian kernels.

Consider two random vectors that will serve as artificial data for this example.

$$\mathbf{x_1} = (z_1, z_2, ..., z_m),\ z_i \sim N(0,1) \text{ i.i.d}$$

$$\mathbf{x_2} = (z_{1'}, z_{2'}, ..., z_{m'}),\ z_{i'} \sim N(0,1) \text{ i.i.d}$$

$$m' = m, \text{ and let } v_i = z_i z_{i'}$$

The expected value of v_i is zero. v_i is the product of two standard normal random variables, which follows an interesting distribution discussed in [12]. The plot of this distribution is shown in figure 2.

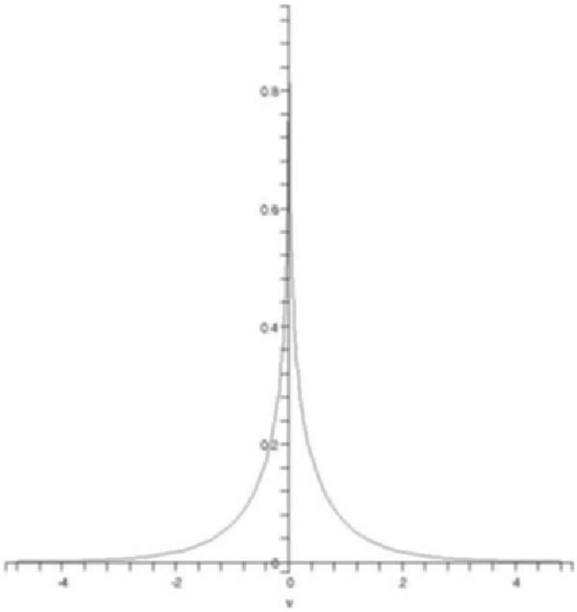


Fig. 2. Plot of $v_i = z_i z_{i'}$

To find the expectation of a linear kernel, it is straightforward to see that $E(\langle \mathbf{x}, \mathbf{y} \rangle) = \sum_i v_i = E(z_1 z_{1'} + z_2 z_{2'} + ... + z_m z_{m'}) = 0$. The variance of the linear kernel can be found as follows:

$$f_{z_i, z_{i'}}(z_i, z_{i'}) \text{ is bivariate normal } \Rightarrow f_{z_i, z_{i'}}(z_i, z_{i'}) = \frac{1}{2\pi} e^{\frac{-(z_i^2 + z_{i'}^2)}{2}}$$

$$f_v(v) = \int_{-\infty}^{\infty} f_{z_i, z_{i'}}(z_i, \frac{v}{z_i}) \frac{1}{|z_i|} dz_i$$

$$E(v) = 0 \Rightarrow \text{ variance } = E(v^2) = \int_{-\infty}^{\infty} v^2 [f_v(v)] \partial v = 1$$

(verified by numerical integration)

Again considering the linear kernel as a function of random variables, $\kappa(\mathbf{x_1}, \mathbf{x_2}) = \langle \mathbf{x_1}, \mathbf{x_2} \rangle = \sum_{i=1}^{m} v_i$ is distributed with a mean of 0 and a variance of $\sum_{i=1}^{m} 1 = m$.

In classification problems, however, it is assumed that the distributions of the variables for one class are not the same as the distributions of the variables for the other class. Let us now consider v_- as a product of dissimilar distributions, and v_+ as a product of similar distributions. Let $v_- = (z_i - 1)(z_{i'} + 1)$. v_- will be distributed with a mean of $\mu_- = E(z_i z_{i'} - z_{i'} + z_i - 1) =$

-1, and a variance of 3 (verified through numerical integration). The linear kernel of the dissimilar distributions can be expressed as:

$$\kappa(\mathbf{x_1} - 1, \mathbf{x_2} + 1) = \sum_{i=1}^{m} v_-$$

This linear kernel is distributed with the following parameters:

$$\text{mean}_- = m\mu_- = -m, \text{ variance } = m\sigma^2 = 3m$$

For the similar observations, let $v_+ = (z_i + 1)(z_{i'} + 1) = (z_i - 1)(z_i' - 1)$. The parameters of the kernel for the similar observations can be found in the same manner. v_+ is distributed with a mean of $\mu_+ = E(z_i z_{i'} + z_{i'} + z_i + 1) = 1$ and a variance of $\sigma^2 = 3$. The linear kernel of the similar distributions can be expressed as:

$$\kappa(\mathbf{x_1} + 1, \mathbf{x_2} + 1) = \sum_{i=1}^{m} v_+$$

This kernel is distributed with the following parameters:

$$\text{mean}_+ = m\mu_+ = m, \text{ variance } = m\sigma^2 = 3m$$

The means and variances of the distributions of the linear kernels are easily tractable, and this is all the information that we need to analyze the effect of dimensionality on these kernels. In the above example, the mean of every variable for dissimilar observations differs by 2. This is consistent for every variable. Obviously, no dataset is this clean, however there are still interesting observations that can be made. Consider that rather than each variable differing by 2, they differ by some value ϵ_i. If ϵ_i is a small value, or even zero for some instances (which would be the case for pure noise), this variable will contribute minimally in distinguishing similar from dissimilar observations, and furthermore the variance of this variable will be entirely contributed. Also notice that at the rate of $3m$, variance grows large fast.

Based on this observation, an assertion is that for the binary classification problem, bimodal variables are desirable. Each mode will correspond to either the positive or negative class. Large deviations in these modes, with minimal variation within a class, are also desired. An effective model must be able to distinguish v_- from v_+. In order for this to occur, the model needs good separation between $mean_-$ and $mean_+$ and variance that is under control.

It is also interesting to explore the gaussian kernel under the same example. For the gaussian kernel, $\kappa(\mathbf{x_1}, \mathbf{y_2}) = e^{-\|\mathbf{x_1} - \mathbf{x_2}\|^2/2\sigma^2}$. This kernel is entirely dependent upon the behavior of $\| \mathbf{x_1} - \mathbf{x_2} \|^2$ and the modeler's choice of the parameter σ (which has no relation to variance).

Restricting our attention to $\| \mathbf{x_1} - \mathbf{x_2} \|^2$, an initial observation is that this expression is nothing more than the euclidean distance squarred. Also, if

$\mathbf{x_1}$ and $\mathbf{x_2}$ contain variables that are distributed $\sim N(0,1)$, then $(\mathbf{x_1} - \mathbf{x_2})$ contains variables distributed normally with a mean of 0 and a variance of 2.

Let $w = (z_i - z_{i'})^2$, implying that $w/2$ is a chi-squarred distribution with a mean of one (which will be annotated as $\chi^2(1)$). This also indicates that $w = 2\chi^2(1)$, indicating that w has a mean of 2 and a variance of 8 (verified by numerical integration).

Therefore, $\| \mathbf{x_1} - \mathbf{x_2} \|^2 = \sum_{i=1}^{m} w_i$ will have a distribution with a mean of $2m$ and a variance of $8m$. Notice that the variance grows much faster under this formulation, indicating even more sensitivity to noisy variables.

The purpose of the above example is to show how every variable added will contribute to the overall behavior of the kernel. If the variable is meaningful, the pattern contributed to the -1 class is not equivalent to the pattern contributed to +1 class. The meaningfulness of the variable can also be considered in terms of cost and benefit. The benefit of including a variable in a classification model is the contribution of the variable towards pushing $mean_-$ away from $mean_+$. The cost of including a variable involves the variance. This variance will be included regardless of the significance of the benefit.

3.3 The Impact of Dimensionality on the One-Class SVM

In order to illustrate the impact of dimensionality on kernels and the one-class SVM specifically, an experiment with artificial data was constructed. This data models a simple pattern involving standard normal distributions where the positive class and negative class have a difference of 2 between their means. This model can be presented as follows:

$$\mathbf{x}_{+1} = (z_1 + 1, z_2 + 1, z_3, ..., z_m),\ z_i \sim N(0,1) \text{ i.i.d}$$

$$\mathbf{x}_{-1} = (z_{1'} - 1, z_{2'} - 1, z_{3'} ..., z_{m'}),\ z_{i'} \sim N(0,1) \text{ i.i.d}$$

The true pattern only lied in the first two variables. All remaining variables were noise. Three types of kernels were examined: the linear kernel, polynomial kernel, and gaussian kernel. Only the results from the gaussian kernel are shown here, however degradation of performance occurred with all kernels. The performance metric used was the area under the ROC curve (AUC).

The gaussian kernels in this experiment were tuned using an auto-tuning method. Typically for gaussian kernels, a validation set of positive and negative labeled data is available for tuning σ. In unsupervised learning, these examples of positive labeled data do not exist. Therefore, the best tuning possible is to achieve some variation in the values of the kernel without values concentrated on either extreme. If σ is too large, all of the values will tend towards 1. If too small, they tend to 0. The auto tuning function ensures that the off-diagonal values for $\kappa(\mathbf{x}_{+1}, \mathbf{x}_{-1})$ average between .4 and .6, with a min value greater than .2.

Table 1. One Class SVM (gaussian kernel) experiment for various dimensions on artificial data

Dimensions	AUC	R^2
2	0.9201	0.5149
5	0.8978	0.4665
10	0.8234	0.4356
50	0.7154	0.3306
100	0.6409	0.5234
250	0.6189	0.4159
500	0.5523	0.6466
1000	0.5209	0.4059

4 A Framework to Overcome High Dimensionality

A novel framework for unsupervised learning, or anomaly detection, has been investigated to solve unsupervised learning problems of high dimension [10,11]. This technique is designed for unsupervised models, however the fusion of model output applies to any type of classifier that produces a soft (real valued) output. This framework involves exploring subspaces of the data, training a separate model for each subspace, and then fusing the decision variables produced by the test data for each subspace. Intelligent subspace selection has also been introduced within this framework.

Combinations of multiple classifiers, or ensemble techniques, is a very active field of research today. However, the field remains relatively loosely structured as researchers continue to build the theory supporting the principles of classifier combinations [18]. Significant work in this field has been contributed by Kuncheva in [16–19]. Bonissone et. al. investigated the effect of different fuzzy logic triangular norms based upon the correlation of decision values from multiple classifiers [4]. The majority of work in this field has been devoted to supervised learning, with less effort addressing unsupervised problems [26]. The research that does address unsupervised ensembles involves clustering almost entirely. There is a vast amount of literature that discusses subspace clustering algorithms [21]. The recent work that appears similar in motivation to our technique include Yang et. al. who develop a subspace clustering model based upon Pearson's R correlation [28], and Ma and Perkins who utilize the one-class SVM for time series prediction and combine results from intermediate phase spaces [20]. The work in this paper has also been inspired by Ho's Random Subspace Methods in [13]. Ho's method randomly selects subspaces and constructs a decision tree for each subspace; trees are then aggregated in the end by taking the mean. Breiman's work with bagging [5] and random forests [6] was also a significant contribution in motivating this work. Breiman's bagging technique involves bootstrap sampling from a training set and creating a decision tree for each sample. Breiman also uses the mean as the aggregator. The random forest technique explores decision tree ensembles from random subsets of features, similar to Ho's method.

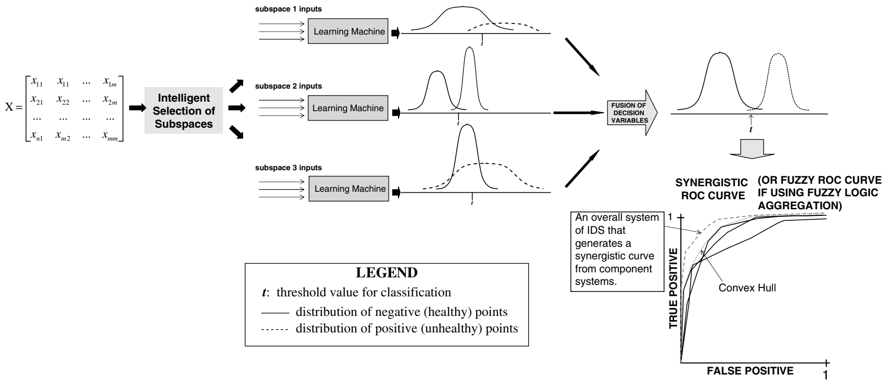

X =
x_{11} x_{11} ... x_{1m}
x_{21} x_{22} ... x_{2m}
x_{n1} x_{m2} ... x_{mm}
Intelligent Selection of Subspaces
subspace 1 inputs
subspace 2 inputs
subspace 3 inputs
Learning Machine
Learning Machine
Learning Machine
t
FUSION OF DECISION VARIABLES
SYNERGISTIC ROC CURVE
(OR FUZZY ROC CURVE IF USING FUZZY LOGIC AGGREGATION)
An overall system of IDS that generates a synergistic curve from component systems.
Convex Hull
TRUE POSITIVE
FALSE POSITIVE
1
1
LEGEND
t: threshold value for classification
distribution of negative (healthy) points
distribution of positive (unhealthy) points

Fig. 3. A sketch of subspace modeling to seek synergistic results.

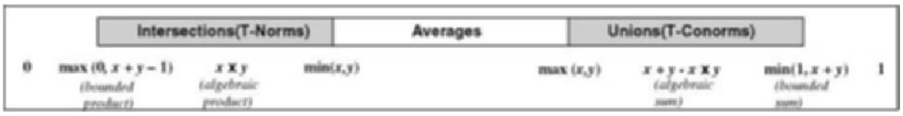


Fig. 4. Aggregation operators

The technique we propose illustrates that unsupervised learning in subspaces of high dimensional data will typically outperform unsupervised learning in the high dimensional data space as a whole. Furthermore, the following hypotheses show exceptional promise based on initial empirical results:

1. Intelligent subspace modeling will provide further improvement of detection beyond a random selection of subspaces.

2. Fuzzy logic aggregation techniques create the fuzzy ROC curve, illustrating improved AUC by selecting proper aggregation techniques.

Promising results from this approach have been published in [10, 11]. As previously discussed, aggregation of models with fuzzy logic aggregators is an important aspect. Given unbalanced data (minority positive class), it has been observed that fusion with T-norms behaves well and improves performance. Figure 4 illustrates the spectrum of fuzzy logic aggregators.

The results shown in table 2 and figure 5 illustrate the improvements obtained through our ensemble techniques for unsupervised learning. The plot of the ROC curves shows the results from using 26 original variables that represented the Schonlau et. al. (SEA) data [9] as one group of variables with the one-class SVM and the result of creating 3 subspaces of features and fusing the results to create the fuzzy ROC curve. It is interesting to notice in the table of results that nearly every aggregation technique demonstrated improvement, especially in the SEA data, with the most significant improvement in the T-norms.

Table 2. Results of SEA data with diverse and non-diverse subsets

	SEA data	**Ionosphere data**
Base AUC (using all variables)	*.7835*	*.931*
T-norms		
minimum	.90	.96
algebraic product	.91	.61
T-conorms		
maximum	.84	.69
algebraic sum	.89	.69

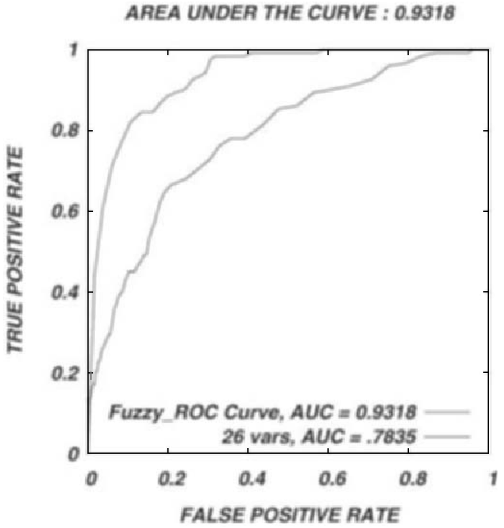


Fig. 5. ROC for SEA data using algebraic product with contention

The ionosphere data is available from the UCI repository, and it consists of 34 variables that represent different radar signals received while investigating the ionosphere for either good or bad structure. For this experiment we again chose $l = 3$.

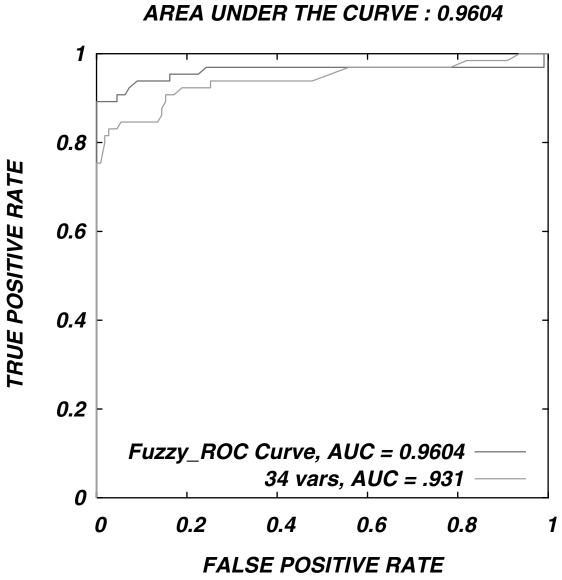


Fig. 6. ROC plot for ionosphere data with minimize aggregation technique

5 Discussion and Conclusion

There were two components to the research presented in this paper. The first component involved exposing the impact of the curse of dimensionality with kernel methods. This involved illustrating that more is not always better in terms of variables, but more importantly that the impact of the curse of dimensionality grows as class imbalance becomes more severe. Kernel methods are not immune to problems involving high dimensional data, and these problems need to be understood and managed.

The second component of this research involved the discussion and brief illustration of a proposed framework for unsupervised modeling in subspaces. Unsupervised learning, especially novelty detection, has important applications in the security domain. This applies especially to computer and network security. Future directions for this research include exposing the theoretical foundations of unsupervised ensemble methods and exploration of other ensembles for the unbalanced classification problem.

References

1. Charu C. Aggarwal and Philip S. Yu. Outlier Detection for High Dimensional Data. Santa Barbara, California, 2001. Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data.
2. Kristin P. Bennett and Colin Campbell. Support Vector Machines: Hype or Hallelujah. 2(2), 2001.

3. Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When Is “Nearest Neighbor” Meaningful? *Lecture Notes in Computer Science*, 1540:217–235, 1999.
4. Piero Bonissone, Kai Goebel, and Weizhong Yan. Classifier Fusion using Triangular Norms. Cagliari, Italy, June 2004. Proceedings of Multiple Classifier Systems (MCS) 2004.
5. Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
6. Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
7. Chih Chung Chang and Chih-Jen Lin. LIBSVM: A Library for Support Vector Machines. http://www.scie.ntu.edu.tw/ cjlin/libsvm, Accessed 5 September, 2004.
8. Yunqiang Chen, Xiang Zhou, and Thomas S. Huang. One-Class SVM for Learning in Image Retrieval. Thessaloniki, Greece, 2001. Proceedings of IEEE International Conference on Image Processing.
9. William DuMouchel, Wen Hua Ju, Alan F. Karr, Matthius Schonlau, Martin Theus, and Yehuda Vardi. Computer Intrusion: Detecting Masquerades. *Statistical Science*, 16(1):1–17, 2001.
10. Paul F. Evangelista, Piero Bonissone, Mark J. Embrechts, and Boleslaw K. Szymanski. Fuzzy ROC Curves for the One Class SVM: Application to Intrusion Detection. Montreal, Canada, August 2005. International Joint Conference on Neural Networks.
11. Paul F. Evangelista, Piero Bonissone, Mark J. Embrechts, and Boleslaw K. Szymanski. Unsupervised Fuzzy Ensembles and Their Use in Intrusion Detection. Bruges, Belgium, April 2005. European Symposium on Artificial Neural Networks.
12. Andrew G. Glen, Lawrence M. Leemis, and John H. Drew. Computing the Distribution of the Product of Two Continuous Random Variables. *Computational Statistics and Data Analysis*, 44(3):451–464, 2004.
13. Tin Kam Ho. The Random Subspace Method for Constructing Decision Forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.
14. Alexander Hofmann, Timo Horeis, and Bernhard Sick. Feature Selection for Intrusion Detection: An Evolutionary Wrapper Approach. Budapest, Hungary, July 2004. International Joint Conference on Neural Networks.
15. Mario Koppen. The Curse of Dimensionality. (held on the internet), September 4-18 2000. 5th Online World Conference on Soft Computing in Industrial Applications (WSC5).
16. Ludmila I. Kuncheva. 'Fuzzy' vs. 'Non-fuzzy' in Combining Classifiers Designed by Boosting. *IEEE Transactions on Fuzzy Systems*, 11(3):729–741, 2003.
17. Ludmila I. Kuncheva. That Elusive Diversity in Classifier Ensembles. Mallorca, Spain, 2003. Proceedings of 1st Iberian Conference on Pattern Recognition and Image Analysis.
18. Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms.* John Wiley and Sons, Inc., 2004.
19. Ludmila I. Kuncheva and C.J. Whitaker. Measures of Diversity in Classifier Ensembles. *Machine Learning*, 51:181–207, 2003.
20. Junshui Ma and Simon Perkins. Time-series Novelty Detection Using One-class Support Vector Machines. Portland, Oregon, July 2003. International Joint Conference on Neural Networks.

21. Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace Clustering for High Dimensional Data: A Review. *SIGKDD Explorations, Newsletter of the ACM Special Interest Group on Knowledge Discovery and Data Mining*, 2004.
22. Vijay K. Rohatgi and A.K.Md. Ehsanes Saleh. *An Introduction to Probability and Statistics*. Wiley, second edition, 2001.
23. Bernhard Scholkopf, John C. Platt, John Shawe Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the Support of a High Dimensional Distribution. *Neural Computation*, 13:1443–1471, 2001.
24. John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
25. Salvatore Stolfo and Ke Wang. One Class Training for Masquerade Detection. Florida, 19 November 2003. 3rd IEEE Conference Data Mining Workshop on Data Mining for Computer Security.
26. Alexander Strehl and Joydeep Ghosh. Cluster Ensembles - A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research*, 3:583–617, December 2002.
27. David M.J. Tax and Robert P.W. Duin. Support Vector Domain Description. *Pattern Recognition Letters*, 20:1191–1199, 1999.
28. Jiong Yang, Wei Wang, Haixun Wang, and Philip Yu. δ-clusters: Capturing Subspace Correlation in a Large Data Set. pages 517–528. 18th International Conference on Data Engineering, 2004.

Part XI

Bioinformatics

A Genetic Algorithm with Self–sizing Genomes for Data Clustering in Dermatological Semeiotics

I. De Falco[1], E. Tarantino[1], A. Della Cioppa[2] and F. Gagliardi[3]

[1] Institute of High Performance Computing and Networking, National Research Council of Italy (ICAR–CNR), Via P. Castellino 111, 80131 Naples – Italy
{ivanoe.defalco,ernesto.tarantino}@na.icar.cnr.it
[2] Dept. of Computer Science and Electrical Engineering, University of Salerno
Via Ponte Don Melillo 1, 84084 Fisciano (SA) – Italy adellacioppa@unisa.it
[3] West–Sud S.r.l., Battipaglia (SA) – Italy francesco.gagliardi@libero.it

Summary. Medical semeiotics often deals with patient databases and would greatly benefit from efficient clustering techniques. In this paper a new evolutionary algorithm for data clustering, the Self–sizing Genome Genetic Algorithm, is introduced. It does not use a priori information about the number of clusters. Recombination takes place through a brand–new operator, i.e., *gene–pooling*, and fitness is based on simultaneously maximizing intra–cluster homogeneity and inter–cluster separability. This algorithm is applied to clustering in dermatological semeiotics. Moreover, a Pathology Addressing Index is defined to quantify utility of the clusters making up a proposed solution in unambiguously addressing towards pathologies.

Key words: genetic algorithms, clustering, semeiotics, variable–length genome

1 Introduction

Semeiotics is the medical discipline which studies signs (symptoms) addressing towards diagnosis of pathologies. One of its goals is detection of syndromes (typical sets of symptoms). A syndrome can be expression of a specific pathology or of totally different pathologies. Detection of a syndrome is important because it addresses next phase, i.e. investigation of underlying physio–pathological causes, with the aim to determine possible pathologies causing syndrome appearance. Moreover syndrome detection is useful in facilitating diagnosis. Semeiotics often deals with patient databases, so it would greatly benefit from efficient data clustering techniques. Yet, the way semeiotics looks at databases is somewhat different from the classical data clustering point of view. In fact, here the aim is to find syndromes which can help doctors by unambiguously addressing towards pathologies, rather than adhering as closely as possible to classes embedded in the database. So, more clusters than classes is not necessarily a drawback, if this helps in reducing ambiguities.

Data clustering [11, 12] is the part of data mining [8, 9] whose task is to group into several subsets (clusters) objects belonging to a given data set, so that members of the same cluster are as similar as possible one another (intra–cluster homogeneity) and as different as possible from members in other clusters (inter–cluster separability). Clustering is performed based on a measure of goodness as the maximization of homogeneity, separability, or both.

Classical data clustering algorithms can be divided into two families. The first is composed by statistical iterative algorithms, like k-mean, k-medioid and k-prototype. The second is composed by hierarchical algorithms, such as the divisive and the agglomerative ones. Typically, the former family obtains solutions in the most effective way, yet with some major limits: the number of clusters is set before run and the final solution depends on the starting point. Algorithms in the latter family do not set cluster number a priori, and let human user evaluate a posteriori the resolution level and the number of clusters into which to partition the assigned data set. Moreover, in both cases there is no explicit simultaneous maximization of homogeneity and separability.

To overcome the above limits we have created an adaptive system based on Genetic Algorithms (GAs) [5, 10], named *Self–sizing Genome Genetic Algorithm* (SGGA). It is self–sizing with respect to the length of genotype encoding the clustering. In our system, the clusters are represented through items called prototypes and search is carried out to simultaneously maximize homogeneity and separability for the clustering.

SGGA has been tested in the medical domain of dermatological semeiotics. Results have been analyzed from two different standpoints. The former is the classical data clustering point of view, so homogeneity and separability values of found solutions have been taken into account. The latter is the semeiotic standpoint, based on which an evaluation index has been defined and used to quantify utility of found syndromes in addressing towards pathologies.

The paper is structured as it follows: firstly, data clustering is presented in Sect. 2, where we introduce the metrics used in the data set space to quantify cluster homogeneity and separability. Section 3 introduces our algorithm. The dermatological semeiotics database faced is shown in Sect. 4, where an evaluation index expressing usefulness of found clusters (representing syndromes) in predicting actual pathologies is also defined. Experiments are described in Sect. 5, and conclusions and planned future works are left to Sect. 6.

2 Data Clustering

2.1 Formalization, Definitions and Notations

Let us denote $\mathbf{S}$ as the universe of all possible elements, said items, with L attributes. Each item is represented by an L-ple: $X \equiv (x^1, ..., x^L)$, where

$X \in \mathbf{S}$ and x^i designates its i–th attribute. Let us consider a data set with cardinality n, i.e. the subset of $\mathbf{S}$ which must be partitioned. Let us denote this set with $\mathbf{DS} = \{X_1, \ldots, X_n\}$, with, in general, $\mathbf{DS} \subseteq \mathbf{S}$.

Distance between Items. For attributes we use the linear distance: $\delta(x^i, y^i) = |x^i - y^i|$, while for items we have chosen the Manhattan distance:

$$d(X,Y) = \frac{1}{L} \times \sum_{i=1}^{L} \left(\frac{1}{R^i} \times \delta(x^i, y^i) \right)$$

where R^i is the range of i-th attribute: $R^i \equiv \delta\big(\max_{j=1,\ldots,n} x^i_j, \min_{j=1,\ldots,n} x^i_j\big)$ (x^i_j is the i–th attribute of the j–th item). $\forall\ X, Y \in \mathbf{DS}$ it results $0 \leq d(X,Y) \leq 1$.

Prototypes and Sharp Clustering for Cluster Representation. We represent a cluster $\mathbf{C_j}$ through an item of $\mathbf{S}$ called prototype P_j which, in general, does not coincide with cluster centroid. A generic item $X_i \in \mathbf{DS}$ is assigned to the cluster $\mathbf{C_k}$ whose representative is the closest. Formally:

$$\text{if } \exists \tilde{k} \ \ : \ d(X_i, P_{\tilde{k}}) \leq d(X_i, P_k) \ \forall k \neq \tilde{k} \ \text{ then } \ X_i \in \mathbf{C_{\tilde{k}}}$$

We call this assignment criterion *sharp clustering* as opposed to fuzzy criteria. A clustering with cardinality m of the data set is a set of m clusters such that: $\mathbf{CL} \equiv \{\mathbf{C_1}, \ldots, \mathbf{C_m}\}$, with $\cup_{k=1}^{m} \mathbf{C_k} = \mathbf{DS}$, and $\forall k, q\ \ \mathbf{C_k} \cap \mathbf{C_q} = \emptyset$.

A clustering can be denoted with the list of the representatives of its clusters: $\mathbf{CL} \equiv \{P_1, \ldots, P_m\}$. Denoting with w_k the cardinality for cluster $\mathbf{C_k}$ we can define the weight array: $W \equiv \{w_1, \ldots, w_m\}$, where $w_i \in \mathbb{N}$ and $\sum_{i=1}^{m} w_i = n$.

Distance between Clusters. If P_i and P_k represent the prototypes for clusters $\mathbf{C_i}$ and $\mathbf{C_k}$, distance between those clusters is defined as:

$$d(\mathbf{C_i}, \mathbf{C_k}) \overset{\text{def}}{=} d(P_i, P_k)$$

2.2 Homogeneity and Separability

Homogeneity for a cluster $\mathbf{C_k}$ is defined as:

$$\mathcal{H}(\mathbf{C_k}) \equiv -\frac{\sum_{X \in \mathbf{C_k}} d(X, P_k)}{w_k} \tag{1}$$

where P_k is the cluster prototype. Hence we can define clustering homogeneity as weighted average of homogeneity of clusters, and separability for a clustering as the weighted average of distances among clusters:

$$\mathcal{H}(\mathbf{CL}) \equiv \frac{\sum_{i=1}^{m} w_i \times \mathcal{H}(\mathbf{C_i})}{\sum_{i=1}^{m} w_i}; \quad \mathcal{S}(\mathbf{CL}) \equiv \frac{\sum_{i=1}^{m-1} \sum_{j=i+1}^{m} w_i \times w_j \times d(\mathbf{C_i}, \mathbf{C_j})}{\sum_{i=1}^{m-1} \sum_{j=i+1}^{m} w_i \times w_j} \tag{2}$$

Due to the definitions of distance, $-1 \leq \mathcal{H}(\mathbf{CL}) \leq 0$, $\ 0 \leq \mathcal{S}(\mathbf{CL}) \leq 1$.

A high value for $\mathcal{H}$ is usually related to a quite high number of clusters, whereas a high value for $\mathcal{S}$ usually means that the clustering has a quite low number of clusters. So, maximization of both has contrasting needs.

3 A Self–sizing Genome Genetic Algorithm

3.1 Genetic Algorithms and Data Clustering

Many GAs applications to data clustering can be found [11]. In a lot of them GAs are linked to traditional clustering algorithms (e.g. k-mean), while pure GAs are usually considered inefficient [11]. To satisfy the constraint of a fixed–length genotype, the encoding usually chosen is the trivial one in which the j–th gene denotes the cluster which the j–th object of data set belongs to.

To limit trivial encoding inefficiency, other encodings have been proposed as in [2] which maps data clustering to Travelling Salesman Problem, or as in [1] where genotype represents the centroids of the clusters. Yet, in all these cases genotype has a fixed length, so we need to know a priori the number of clusters. Another drawback of the above approaches is that clustering quality is evaluated in terms of homogeneity only. To overcome these limitations, our GA follows some basic concepts present in messy GAs [6], and, like them, can handle variable–length genotypes, yet it has features typical of its own. Moreover, the fitness function takes into account also separability.

3.2 SGGA for Data Clustering

SGGA aims at finding the number of clusters and their prototypes by using a genotype similar to that in [1] and [4], but with variable length. Once found the prototypes, data set segmentation is achieved with sharp clustering.

Genotype. A genotype is a list (with variable length) of chromosomes (each encoding a prototype):

$$\mathbf{G} \equiv \{(P_1; \ldots ; P_\ell)\} \equiv \{(y_1^1, \ldots , y_1^L); \ldots ; (y_\ell^1, \ldots , y_\ell^L)\}$$

where P_j is the prototype of j–th cluster, y_j^i is the i–th attribute of j–th cluster and ℓ is not set a priori. Any attribute is represented by values of same type as attribute in its own variation range. Order of prototypes in the genotype is not important: all genotypes achievable from a given one by permuting its chromosomes express the same solution.

Clustering. Data set items are assigned to clusters based on sharp clustering criterion with respect to prototypes, as seen above. Homogeneity of clustering represented by genotype $\mathbf{G}$ can then be evaluated by means of (2). Distance between clusters is computed as the distance between respective prototypes, so separation of clustering represented by $\mathbf{G}$ is computed through (2).

Fitness Function. After performing data set segmentation based on a genotype $\mathbf{G}$, its quality must be evaluated by using a fitness function f. Choice of f is critical especially as concerns number of clusters $\mathcal{N}$. In fact, to avoid possible solution overfitting to the data set due to uncontrolled increase in $\mathcal{N}$, we might add in the fitness a penalty term proportional to the number of clusters. Yet, this would introduce an upper limit to $\mathcal{N}$, by using a priori knowledge on the application domain. To get rid of this problem, following [14], we define:

Algorithm 1 Gene–Pooling Algorithm

```
begin
    create genetic soup $\mathbf{GS}_{(\ell_1+\ell_2)} = \mathbf{G}^1_{\ell_1} \cup \mathbf{G}^2_{\ell_2}$;
    eliminate duplicate chromosomes in $\mathbf{GS}_{(\ell_1+\ell_2)}$ thus obtaining $\mathbf{GS}_{\tilde{\ell}}$;
    order chromosomes in $\mathbf{GS}_{\tilde{\ell}}$ based on weight;
    randomly choose number $\ell$ of chromosomes for the offspring, $\ell \in [2, \ell_1 + \ell_2]$;
    give the first $\ell$ prototypes of $\mathbf{GS}_{\tilde{\ell}}$ to offspring genotype $\mathbf{G}_\ell$;
end
```

$$f(\mathbf{G}) = \mathcal{H}(\mathbf{G}) + \mu \times \mathcal{S}(\mathbf{G})$$

where μ represents a scaling factor ($\mu \geq 0.0$). The following limit cases exist: $\mu \to 0 \Rightarrow f \to \mathcal{H}(\mathbf{G}) \Rightarrow \mathcal{N} \to +\infty$ and $\mu \to +\infty \Rightarrow f \to \mu \times \mathcal{S}(\mathbf{G}) \Rightarrow \mathcal{N} \to 1$. So data clustering is formulated as a problem of direct maximization for $\mathcal{H}$ and $\mathcal{S}$ independently of $\mathcal{N}$. Given the above ranges for $\mathcal{H}(\mathbf{G})$ and $\mathcal{S}(\mathbf{G})$, it turns out that $-1 \leq f(\mathbf{G}) \leq \mu$.

Selection. Tournament method has been chosen. We denote with $\mathcal{T}$ its size.

Gene-pooling Operator. Given two parents, this operator creates an offspring with a variable number of chromosomes chosen among those contained in both parents. It is applied deterministically with a probability of 100%. *Gene-pooling* is similar to cut'n'splice operator defined in messyGA and can be seen as an improvement of the homonymous operator presented in [4] in that, differently from this latter, it does not impose limits on maximal and minimal length of offspring. If $\mathbf{G}_\ell$ is a genotype of length ℓ then gene–pooling operator is of the type: $(\mathbf{G}_{\ell_1}, \mathbf{G}_{\ell_2}) \longrightarrow \mathbf{G}_\ell$, with $\ell \in [2, \ell_1 + \ell_2]$.

Probability distribution for offspring genotype length $P(\ell)$ is defined as:

$$P(\ell) = \begin{cases} \frac{1}{\ell_1+\ell_2-1} \text{ for } \ell \in [2, \ell_1 + \ell_2] \\ 0 \text{ otherwise} \end{cases}$$

This operator does not depend on the order of the prototypes in the parents, and selects deterministically the chromosomes which the offspring will inherit. Selection takes into account weights which chromosomes, seen as prototypes, have in parents genotypes: the heavier a prototype (i.e. the higher its cardinality), the more likely it will be present in the offspring. Given two genotypes $\mathbf{G}^1_{\ell_1}$ and $\mathbf{G}^2_{\ell_2}$, Algorithm 1 shows gene–pooling procedure.

Offspring can have a number of chromosomes up to the sum of ℓ_1 and ℓ_2, like cut'n'splice operator and unlike gene–pooling defined in [4].

Mutation Operator. It acts on prototype attributes of the offspring achieved by gene–pooling, and modifies them with a probability p_m. This probability is normalized to whole genotype length $\ell \times L$, where ℓ denotes genotype length in terms of chromosomes and L is the number of attributes for each prototype. To avoid that genotypes with more chromosomes are more exposed to mutations, we set: $p_m = \frac{\eta}{\ell \times L}$, where η is said mutation factor and represents the average number of mutations each genotype undergoes. The specific mutation mechanism used depends on types of attributes in the database.

Table 1. Pathologies contained in the data set

class	pathology	no. of instances
1	psoriasis	112
2	seboreic dermatitis	61
3	lichen planus	72
4	pityriasis rosea	49
5	chronic dermatitis	52
6	pityriasis rubra pilaris	20

Population Initialization. Genotype lengths of the initial population are chosen in a uniform random way in the range $[\mathbf{G}_{\ell_{\mathbf{min}}}, \mathbf{G}_{\ell_{\mathbf{max}}}]$, where these two values are parameters for the algorithm. Attribute values for any prototype are generated randomly with uniform probability. One more parameter, $\mathbf{G}_{\ell_{\mathbf{dyn}}}$, is the upper bound for genotype length during evolution.

4 Data Set and Pathology Addressing Index

4.1 The Considered Data Set

We have used the "dermatology" database [3] which contains clinical dermatological cases. It has been chosen because it also presents classification into pathologies, so as to evaluate a posteriori quality of found syndromes. It consists of 366 clinical cases, each with 34 attributes. Any attribute, apart from age and family anamnesis, is expressed by an integer value in $[0, 3]$ where 0 means that the attribute is not present, 3 states that it is present at its maximum degree, 1 and 2 denote intermediate values. Family anamnesis is a boolean attribute, has value 1 if any pathology has been observed in the family, 0 otherwise. Pathologies present in the data set are six (see Table 1).

Classes 2 and 4 are very similar, even medical experts meet problems in distinguishing between them. Moreover, neither clustering methods based on Neural Networks with voting features endowed with GAs [7] nor tree–structured self–organizing Neural Networks [13] get rid of this problem.

4.2 Pathology Addressing Index

Given a database of clinical charts, each item belongs to only one of the p pathologies (classes) based on medical opinion contained in the database. By sharp clustering, each item is assigned to only one of s syndromes (represented by a cluster). Then a table between s syndromes and p pathologies can be built that corresponds to a matrix $\mathbf{A}$, where element a_{ij} is the number of elements in the data set assigned to syndrome j which are affected by pathology i.

Based on this matrix we define a Pathology Addressing Index $\mathcal{I}$ which can be useful from a semeiotic standpoint to evaluate the degree of utility of found syndromes in unambiguously addressing towards pathologies.

Firstly, we define $\mathcal{I}$ for the generic j-th syndrome as it follows: $\mathcal{I}^j = \sum_{i=1}^{p} \frac{a_{ij}}{\rho^j}$, where $\rho^j = \max\limits_{i \in \{1,\ldots,p\}} a_{ij}$. This index is 1 in the best case, in which the syndrome represents only one pathology, and is p in the worst case, in which the syndrome addresses uniformly towards all considered pathologies.

Then, we define $\mathcal{I}$ for the whole clustering as the weighted average for $\mathcal{I}^j$:

$$\mathcal{I} = \frac{\sum_{j=1}^{s} w^j \times \mathcal{I}^j}{\sum_{j=1}^{s} w^j}$$

where w^j denotes the weight of j–th syndrome, i.e. the number of clinical cases assigned to j–th cluster. $\mathcal{I}$ can vary within 1 and p, and the higher its value the more ambiguously on average each syndrome addresses towards $\mathcal{I}$ pathologies rather than towards one. $\mathcal{I}$ usually tends to 1 as s increases.

5 Experimental Results of SGGA

As the given database contains integer values only, any attribute in any genotype can take on integer values in its own variation range. Any such value is directly represented as an integer. Mutation randomly changes any such value into one among the allowable ones in attribute range, with uniform probability. In all experiments we have used the following parameter setting: $\eta = 0.05$, $\mathcal{T} = 10$, $\mathbf{G}_{\ell_{\mathbf{min}}} = 2$, $\mathbf{G}_{\ell_{\mathbf{max}}} = 14$, $\mathbf{G}_{\ell_{\mathbf{dyn}}} = 50$, population size $p_s = 100$ and maximum number of generations $n_g = 300$. Since the given database has 6 pathologies, range for $\mathcal{I}$ is $1 \leq \mathcal{I} \leq 6$. We have taken into account for μ the set of values in the range $[0.0, 1.0]$ with step 0.1. For each such value our algorithm has been run ten times with different random seeds.

We have performed two types of analysis, one in terms of canonical data clustering, and the other from the semeiotic point of view.

5.1 SGGA Performance as a Data Clustering Tool

Table 2 reports the final results averaged over the 10 runs in terms of $\mathcal{H}$, $\mathcal{S}$, $\mathcal{N}$ and $\mathcal{I}$. It can be seen that as μ increases $\mathcal{H}$ decreases while trend of $\mathcal{S}$ is increasing, as expected. Indirect control on $\mathcal{N}$ is achieved as μ varies, as the decreasing trend in Table confirms. Moreover, $\mathcal{I}$ increases with μ (we remind that higher $\mathcal{N}$ values are expected to be related to lower $\mathcal{I}$ values).

Table 2 can be broadly divided into three parts.

For high values of μ (≥ 0.90), system behavior is not good. In fact, homogeneity values are quite lower than in other cases. In our opinion, this is due to the fact that for any obtained clusterization values for separability are very often quite higher (about two or three times in magnitude) than those for

Table 2. Average values for $\mathcal{H}$, $\mathcal{S}$, $\mathcal{N}$ and $\mathcal{I}$ as a function of μ

μ	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	1.00
$\mathcal{H}$	**-0.099**	-0.100	-0.103	-0.102	-0.104	-0.105	-0.110	-0.118	-0.119	-0.129	-0.131
$\mathcal{S}$	0.166	0.166	0.188	0.186	0.198	0.201	0.209	0.222	0.221	0.230	**0.238**
$\mathcal{N}$	26.2	22.5	19.6	16.3	14.4	13.6	10.8	7.6	5.8	5.4	4.6
$\mathcal{I}$	1.239	1.210	1.384	1.327	1.491	1.524	1.726	1.806	1.836	1.891	2.020

homogeneity, and the former is not downsized by the value of μ which is close to or equal to 1.0. Thus evolution is driven by the former parameter. This causes the program to create just few clusters (3–5), as expected. This, on its turn, determines bad homogeneity within any cluster. Therefore, μ values should be not too high if good–quality clusterings are sought.

For low values of μ (≤ 0.10), instead, evolution is strongly based on homogeneity, which is the only active part of fitness for $\mu = 0$. As expected, many clusters (about 20–25) are formed with this setting, and $\mathcal{H}$ values are good (about -0.100 on average), while $\mathcal{S}$ is quite low (about 0.166 on average), resulting in quite bad clusterings. This leads us to consider that separability must be present in the fitness function, so μ should be greater than zero.

Better clusterings are found for $0.20 \leq \mu \leq 0.80$, where system is able to balance the two contrasting needs of maximizing $\mathcal{H}$ and $\mathcal{S}$ at the same time.

Comparison about solutions found for different values of μ (i.e., for different fitness landscapes) cannot be carried out in terms of fitness values. Rather, since Table 2 shows that as $\mathcal{H}$ decreases, $\mathcal{S}$ increases and vice versa, comparison can be effected in terms of proximity to an ideal case. We consider as ideal situation the one represented by the point P having as coordinates the highest values for $\mathcal{H}$ and $\mathcal{S}$ in Table 2. P has coordinates $\mathcal{H}=-0.099$, $\mathcal{S}=0.238$. By evaluating distances of any of the above ($\mathcal{H}$, $\mathcal{S}$) pairs from P, we find that the best clusterizations are achieved, on average, for $\mu = 0.70$.

Figure 1 shows typical SGGA behavior. In it we report the best run (in terms of fitness) for $\mu = 0.70$. Left part shows fitness evolution: both best and average fitnesses increase over generations. Right part shows behavior of $\mathcal{N}$ as a function of number of generations. During the very first generations (up to generation $10 - 15$) the value of $\mathcal{N}$ increases, then in the next phase (generations 15 to 40) the system is able to reverse this trend and to reduce genotype lengths. After about 40 generations the population comes to an almost stationary phase as regards $\mathcal{N}$, and in next 260 generations no significant variations appear, especially as regards average value of $\mathcal{N}$.

5.2 Comparison of Found Syndromes with Known Pathologies

From a semeiotic standpoint, evaluation of results is quite different from the previous one based on pure data clustering considerations. In fact, Table 2 shows that $\mathcal{I}$=1.806 on average for $\mu = 0.70$, which reveals some ambiguity

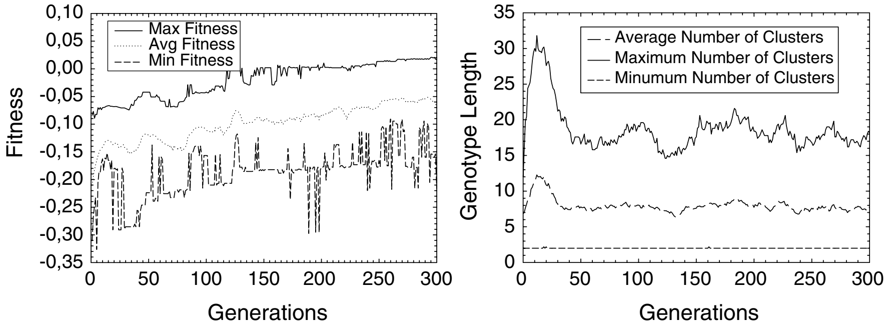


Fig. 1. A typical evolution of our system: (**left**) fitness; (**right**) number of clusters

Table 3. Syndromes (s) – pathologies (p) table for the best solution for $\mu = 0.30$

$p\backslash s$	1	2	3	4	5	6	7	8	9	10	11	12	13
1	1	0	10	0	0	25	0	28	25	0	0	19	0
2	30	13	0	0	0	0	8	0	0	0	23	0	0
3	0	1	0	47	0	0	0	0	0	24	0	0	0
4	1	7	0	0	0	0	29	0	0	0	1	0	0
5	2	2	0	0	47	0	1	1	1	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	20

in addressing capability of achieved syndromes. Rather, a semeiotician would choose as best situation the one with the lowest $\mathcal{I}$. This holds at $\mu = 0.10$, yet all μ values in $[0.0, 0.30]$ provide good values for $\mathcal{I}$. A reasonable trade–off between $\mathcal{I}$ and a non–excessive number of found syndromes might be $\mu = 0.30$.

Table 3 shows the relations between syndromes (s) and pathologies (p) for the clustering obtained in the best (in terms of fitness) of the runs for $\mu = 0.30$. Seven out of the thirteen found syndromes indicate univocally a pathology. This yields a very low $\mathcal{I}$ value (1.101). Syndrome 13 contains all items of pathology 6, and contains only them, so it correctly addresses towards pityriasis rubra pilaris. Syndromes 3, 6 and 12, instead, correctly address exclusively towards pathology 1. Syndromes 4 and 10 address exclusively towards pathology 3, and so does syndrome 5 towards pathology 5. Remaining syndromes are ambiguous, since each of them addresses towards at least two pathologies.

As regards pathologies, our algorithm has problems in distinguishing between pathologies 2 and 4, as is the case for other algorithms: syndromes 1, 2, 7 and 11 contain examples from both. Nonetheless they can help doctors: in fact, among them 1 and 11 most likely address towards pathology 2 whereas 7 most likely addresses towards pathology 4, and only 2 is actually ambiguous. Pathology 6 is totally addressed by syndrome 13, and pathology 5 is almost completely addressed by syndrome 5. Pathology 3 is mostly addressed by syndromes 4 and 10, while pathology 1 is addressed by six different syndromes.

6 Conclusions and Future Works

In this work we have devoted our attention to unsupervised syndrome discovery, and we have realized a Scientific Discovery Support System which can, at least in principle, discover unknown syndromes. The implemented system is actually a cognitive artificial system able to substitute the doctor in detecting syndromes orienting towards pathologies. As a data clustering algorithm, SGGA has satisfied expectations, reaching the initial goals of not requiring a priori knowledge about faced problem. In fact, it has not been necessary to introduce the number of clusters in the fitness function.

As regards future works, we plan to use SGGA to face a wide set of clustering problems in medical domain. Furthermore, we intend to compare (in terms of $\mathcal{I}$ index) the quality of solutions achieved by our system against those provided by methods typically used in clustering, like for instance k–means.

References

1. Babu GP, Murty MN (1993) A near–optimal initial seed value selection in k-means algorithm using a genetic algorithm, Pattern Recogn. Lett. 14(10):763–769
2. Bhuyan J, Raghavan V, Venkatesh K (1991) Genetic algorithm for clustering with an ordered representation. In: Belew RK, Booker LB (eds) Proc. of the Fourth Int. Conf. on Genetic Algorithms (1991), pp. 408–415. Morgan–Kaufmann San Mateo
3. Blake CL, Merz CJ (1998) UCI Repository of machine learning databases: University of California, Irvine [http://www.ics.uci.edu/∼mlearn/MLRepository.html]
4. Burdsal B, Giraud–Carrier C (1997) Evolving fuzzy prototypes for efficient data clustering, www.cs.bris.ac.uk/Tools/Reports/Ps/1997-burdsall.ps
5. Goldberg DE (1989) Genetic algorithms in search optimization and machine learning. Addison–Wesley, Reading, Mass
6. Goldberg DE, Korb B, Deb K (1989) Messy genetic algorithms: motivation, analysis, and first results, Complex Systems 3:493–530
7. Guvenir HA, Demiroz G, Ilter N (1998) Learning differential diagnosis of erythemato–squamous diseases using voting feature intervals, Artificial Intelligence in Medicine 13:147–165
8. Han J, Kamber M (2001) Data mining: concept and techniques. Morgan Kaufman
9. Hand DJ, Mannila H, Smyth P (2001) Principles of data mining. MIT Press
10. Holland JH (1975) Adaptation in natural and artificial systems, 2nd edition. The University of Michigan Press
11. Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review, ACM Computing Surveys 31(3):264–323
12. Kaufman L, Rousseeuw PJ (1990) Finding groups in data. An introduction to cluster analysis. Wiley and Sons, New York
13. Luo F, Khan L, Yen IL, Bastani F (2003), A dynamical growing self–organizing tree (DGSOT) for hierarchical clustering. Submitted to IEEE Transactions on Knowledge and Data Engineering, July, 2003. utdallas.edu/ luofeng/DGSOT.doc
14. Yip AM (2002) A scale dependent data clustering model by direct maximization of homogeneity and separation. In: Proc. Mathematical Challenges in Scientific Data Mining IPAM. 14–18 January (2002)

MultiNNProm: A Multi-Classifier System for Finding Genes

Romesh Ranawana, Vasile Palade

University of Oxford, Computing Laboratory,
Oxford, OX12HY, United Kingdom

Abstract. The computational identification of genes in DNA sequences has become an issue of crucial importance due to the large number of DNA molecules being currently sequenced. We present a novel neural network based multi-classifier system, MultiNNProm, for the identification of promoter regions in E.Coli[1] DNA sequences. The DNA sequences were encoded using four different encoding methods and were used to train four different neural networks. The classification results of these neural networks were then aggregated using a variation of the LOP method. The aggregating weights used within the modified LOP aggregating algorithm were obtained through a genetic algorithm. We show that the use of different neural networks, trained on the same set of data, could provide slightly varying results if the data were differently encoded. We also show that the combination of more neural classifiers provides us with better accuracy than the individual networks.

Keywords: Multi-Classifier Systems, Neural-Networks, E.Coli, Promoters, Genetic Algorithms

1 Introduction

The recognition of genes in strings of DNA has become an important issue due to the vast amount of gene sequences being currently produced around the world. Once a DNA molecule has been sequenced, it is necessary to first identify the genes and protein coding regions within the sequence, and then understand their functionality. It is both economically and practically impossible for these steps to be conducted manually, and thus, the development of accurate and efficient computational algorithms for the recognition of genes in DNA sequences has become an urgent necessity.

It is impractical to design traditional algorithms for the recognition of genes within these sequences due to the complexity of the task at hand. The main goal of gene classification is generalization, that is, being able to induce a concept that accurately classifies genes not included in the training set. The difficulty with achieving good generalization capability lies in the fact that a learner cannot directly measure the generalization ability; and instead, the learner is forced into relying on its inductive bias to hopefully produce an accurate classifier. A proper classification system should ideally be able to identify genes which were not included in the training set. Machine learning approaches, therefore, offer promising solutions to solve complex problems, due to their capabilities in identifying patterns and genetic concepts automatically. Some of the more successful examples of gene recognition methods include the use of neural networks (Snyder and Stormo, 1995; Uberbacher and Mural, 1991; Ma et al., 2002), Decision Trees (Salzberg et al., 1997), Genetic Programming (Koza and Andre, 1995), Hidden Markov Models (Henderson et al., 1997; Kulp et al., 1996; Krogh, 1996; Birney, 2001) and Fuzzy Logic (Ohno-Machado et al., 1998; Woolf and Yang., 2000).

Promoter regions are known to be situated immediately before genes. Thus, the successful identification of a promoter leads to the identification of the starting position of a gene. We propose a neural network based multi-classifier system, MultiNNProm, for the identification of these promoter sites in E.Coli[1] DNA sequences. The proposed system contains four neural networks, to which the same DNA sequence is presented using four different encoding methods. The outputs of the individual neural networks are then passed through a probability function and finally aggregated by a combination function. This combinatorial function then accumulates the results to provide an answer as to whether the presented sequence is an E.Coli promoter or not.

2 Biological Background

DNA, deoxyribonucleic acid, is found in long chains, with each link called a nucleotide. Each nucleotide in DNA consists of a sugar molecule called deoxyribose that bonds to a phosphate molecule and to a nitrogen containing compound, known as a base. DNA consists of four bases in its structure: adenine(A), cytosine(C), guanine(G) and thymine(T). The order of the bases in a DNA molecule – the genetic code – determines the amino acid sequence of a protein. Genes are actually a subset of a cell's DNA. While every gene is made up of a DNA sequence, the entire DNA of an individual is not made up of genes. Indeed, only two to three percent of a humans DNA is made up of genes.

The Bioinformatics Initiative[2] lists out a set of reasons why E.Coli is currently the organism of choice for many research studies, both as a reference organism for many studies on prokaryotic systems and as a source of information on proteins and metabolic pathways that are shared by eukaryotes as well. Ma et al., (2002) used the software available at http://www-lecb.ncifcrf.gov/~toms/delia.html to display the sequence logos of 438 E.Coli promoters that were aligned according to

[1] Escherichia coli, a species of bacterium normally present in intestinal tract of humans and other animals; sometimes pathogenic; can be a threat to food safety

[2] http://www.ecoli.princeton.edu/E_coli_Bioinformatics_Document.pdf

their transcriptional start sites[3]. These results are displayed in Fig. 1. The software used the Shannon Entropy to independently measure the non-randomness of each position. From Fig. 1, it can be seen that strong motifs[4] exist at position -35 and -10, and weak motifs exist at positions +1, -22, -29 and -44. A weak motif is a sequence of bases present in DNA which is not as statistically significant as a strong motif, where the probability of a certain consensus sequence appearing within every gene or promoter is much higher. The difference between these weak and strong motif's can be identified by observing the height of the peaks at each position in Fig. 1.

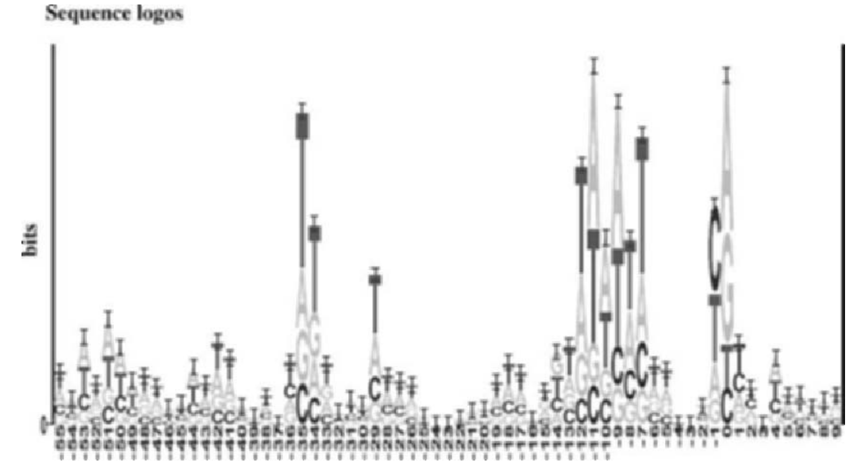


Fig. 1. The sequence logos of 438 E.Coli promoters

3 Why use Multiple Classifiers?

Individual classifier models are recently being challenged by combined pattern recognition systems, which often show better performances than individual classifiers. Thus, when a system that incorporates many methodologies is used, the inadequacies of one methodology can usually be nullified by the advantages offered by another methodology. Thus, genes not recognized by one classifier will be recognized by another, and DNA falsely classified as being genes by one

[3] The start position of where the organic process whereby the DNA sequence in a gene is copied into mRNA.

[4] A conserved element of a protein sequence alignment that usually correlates with a particular function. Motifs are generated from a local multiple protein sequence alignment corresponding to a region whose function or structure is known. It is sufficient that it is conserved, and is hence likely to be predictive of any subsequent occurrence of such a structural/functional region in any other novel protein sequence

classifier will be rejected by another. The challenge in building such a system lies in the design and implementation of a combinatorial function which combines the results provided by each classifier in order to come up with the optimal result. Here, the performance of the entire system will be proven to be never worse than that of the best individual expert. It is a well-known fact that, if a multi-classifier system is to be successful, the different classifiers should have good individual performances and be sufficiently different from each other (Sharkey and Sharkey, 1997). But, neither individual performances (Rogova, 1994; Zenobi and Cuningham, 2001), nor the diversity (Ruta and Gabrys, 2001) provides an ideal measure of how successful the combination of the classifiers will be.

It has been proven that the usage of an ensemble of neural networks for a certain classification problem can improve the classification performance when compared to the use of singular neural networks. (Rost and Sander, 1993) and (Riis and Krogh, 1996) provide results tested on protein secondary structure prediction. (Baldi and Brunak, 1998) also lists out an overview of applications in molecular biology. Among different methodologies used for combining multiple neural network classifiers, the majority voting, neural networks, Bayesian inference and the Dempster-Shafer theory have been proven to be the most popular (Mandler and Schurmann, 1988; Xu et al., 1991 and 1992). The Dempster-Schafer method had been proven to be successful, but has a considerable dependency on the function used for the alignment of the probability. For the type of output produced by neural networks, posterior class-conditional probabilities can be calculated. The calculation of these probabilities becomes relatively simple, especially when the number of output classes is small.

In the next section of the paper, we present two methods that can be used for the combination of results obtained through multiple neural network classifications. These two methods, the LAP (Linear Average Predictor) and LOP (Logarithmic Opinion Pool) methods, were introduced in (Hansen and Krogh, 1999). We propose a modification of the LOP method - LOP2 - which was proven to produce better results on the data set used.

3.2 The LAP and LOP Methods for Combining Classifiers

The method that we used for the combination of our four classifiers is a variation of the LOP method (Hansen and Krogh, 1999), which we name LOP2. We compare this method with the LAP method, which is more commonly used. The LOP method is a general ensemble method which defines a method on how to calculate the error of an ensemble (called the ensemble ambiguity) using the individual errors of different classifiers. Hansen and Krogh claimed that ensembles always improve the average performance.

An ensemble consists of M predictors f_i, which are combined into the combined predictor F. Let each predictor produce an output in the form of a probability vector $\{f_i^1,...,f_i^N\}$, where f_i^j is the estimated probability that the input $\vec{x}$ belongs to the class c_j. There is also a vector of coefficients, of the form $\{\alpha_1,...,\alpha_M\}$, associated with each ensemble, where $\sum_i \alpha_i = 1,\ \alpha_i \geq 0$.

3.2.1 The LAP Method

The combined predictor for the LAP method is defined as:

$$F_{LAP}^{j} = \sum_{i=1}^{N} \alpha_i f_i \qquad \textbf{(1)}$$

3.2.2 The LOP Method

The combined predictor for the LOP method is defined by equation (2), where Z is the normalization factor defined by equation (3). This method is non-linear and asymmetric in comparison to the LAP method. A comparison of the results obtained using the two methods is given in section 5.

$$F_{LOP}^{j} = \frac{1}{Z} \exp\left(\sum_{i=1}^{M} \alpha_i \log f_i^{\,j} \right) \qquad \textbf{(2)}$$

$$Z = \sum_{j=1}^{N} \exp\left(\sum_{i=1}^{M} \alpha_i \log f_i^{\,j} \right) \qquad \textbf{(3)}$$

3.2.3 The LOP2 Method

The LOP combination method was changed by removing the condition $\sum_{i=1}^{M} \alpha_i = 1$, with α_{i} being allowed to take on negative values. We found that this small modification allowed the system to nullify small errors created by the more successful classifiers.

4 Description of the MultiNNProm System

4.1 Overview of the System

As shown in Fig. 2, the system was implemented as a neural-network based multi-classifier system. We developed four neural networks, NNE1, NNE2, NNC2 and NNC4, into which the same DNA sequence was inputted using four different encoding methods: E1, E2, C2 and C4 (see Table 1), respectively. The outputs of the individual neural networks are then passed onto a probability builder function which assigned probabilities as to whether the presented sequence was an E.Coli promoter or not. Finally, the outputs of the probability functions are aggregated by a result combiner, which combined the results and produced a final result as to whether the given sequence was an E.Coli promoter or not. The final output was of the 'yes' or 'no' form.

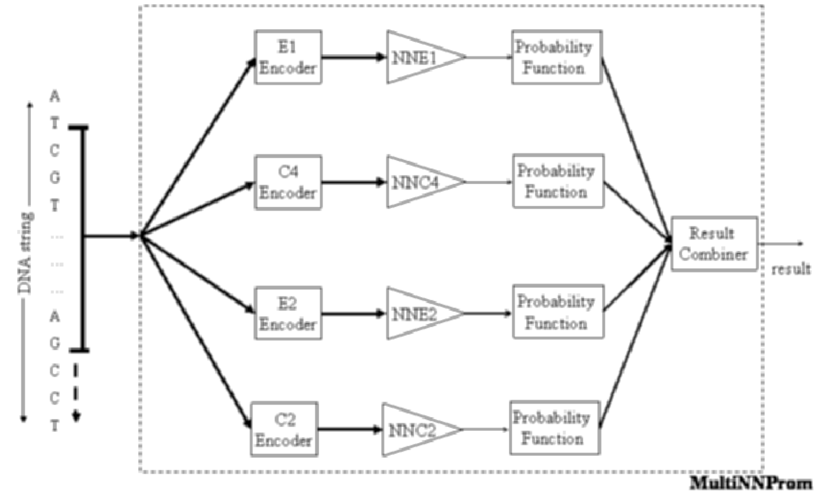


Fig. 2. MultiNNProm Architecture

Table 1. The input encoding methods used in the implementation of the system

Encoding Method	Originally used by	A	T	C	G
E1		-2	-1	1	2
E2		-1	-1	1	1
C2	Demeler and Zhou, 1991	00	01	11	10
C4	Brunak et al., 1991	1000	0100	0010	0001

The entire dataset consisted of 471 E.Coli promoter sequences and 600 non-promoter sequences. The positive data sequences were obtained from the PromEC[5] website. Each of the obtained E.Coli sequences were 101 nucleotides in length and were aligned at the transcriptional start site. Each of the transcriptional start sites were positioned such that they each appeared at position 76 of the string. Thus, each of the positive sequences consisted of 101 nucleotides starting from the -75 position, more exactly 75 nucleotides upstream of the transcriptional start site, and ending at the +25 position, which is 25 nucleotides downstream of the transcriptional start site. The negative data set was obtained from the database compiled by the E.Coli GenomeProject, at the University of Wisconsin-Madison[6]. The negative dataset consists of E.Coli genes with the preceding promoter region deleted. Using the given set of available data, we constructed three data sets: a training sequence, a positive test sequence and a negative test sequence. The training set consisted of 324 positive sequences and 429 negative sequences. The remaining positive and negative data were divided up into a positive test set of 159 sequences and a negative test set of 171 sequences.

4.2 The Probability Function

The probability function assigns a probability as to whether a given sequence is an E.Coli promoter or not. The same probability function is used for the output of each neural network. The probability functions used to calculate the probabilities for positives and negative data are defined in equations (4) and (5).

$$Positive(i) \;=\begin{cases} 0.999 \;;\; i \geq 1 \\ 0.001 \;;\; i \leq -1 \\ \dfrac{i+1}{2} \;;\; else \end{cases} \tag{4}$$

$$Negative(i) =\begin{cases} 0.001 \;;\; i \geq 1 \\ 0.999 \;;\; i \leq -1 \\ \dfrac{1-i}{2} \;;\; else \end{cases} \tag{5}$$

If the output of the system was 1 or more than one, we deemed the given sequence to be an E.Coli promoter. If the output of the system was -1 or less than -1, the conclusion was that the given sequence was not an E.Coli promoter. We assigned a probability of 0.999 to either the ‘Yes’ value or the ‘No’ value in such a case, because of the use of the logarithm within the result combiner.

[5] http://bioinfo.md.huji.ac.il/marg/promec/

[6] http://www.genome.wisc.edu/sequencing/k12.htm#seq

4.3 Result Aggregation

The outputs of the four neural networks were combined using the LOP2 method described in Section 3.2. The system was tested on three combination methods, namely, classic majority voting, LAP, and LOP2. The experimental observations showed us that the results obtained through the LOP2 method provided us with a much better recognition rate for the test data, with respect to both positive and negative data. The comparison between these three aggregation methods is listed out in Section 5.

The LOP2 method was implemented as follows. Let the outputs of the four neural networks be symbolized by O_i, where $1 \le i \le 4$. We also defined four coefficients α_i, where $1 \le i \le 4$. Then, the combined predictor is defined by equations (6), (7) and (8).

$$Positive(O) = \frac{1}{Z}\exp\left(\sum_{i=1}^{4}\alpha_i\left(\log\left(Positive(O_i)\right)\right)\right) \quad \textbf{(6)}$$

$$Negative(O) = \frac{1}{Z}\exp\left(\sum_{i=1}^{4}\alpha_i\left(\log\left(Negative(O_i)\right)\right)\right) \quad \textbf{(7)}$$

$$Z = \exp\left(\sum_{i=1}^{4}\alpha_i\left(\log\left(Positive(O_i)\right)\right)\right) + \exp\left(\sum_{i=1}^{4}\alpha_i\left(\log\left(Negative(O_i)\right)\right)\right) \quad \textbf{(8)}$$

It is obvious that $Positive(O) + Negative(O) = 1$. The final conclusion on whether the given sequence was an E.Coli promoter or not was reached using equation (9). A Genetic Algorithm was used for the determination of the aggregation coefficients. Ten initial populations of random coefficients were initialized and run through a genetic algorithm for 60 generations. The best performing chromosomes of each population were then extracted and run through another genetic algorithm. The fitness function utilized for the selection of chromosomes was the classification error exhibited by the multi-classifier system. This function calculated the percentage of erroneous identifications with respect to the total number of testing data presented. Thus, the genetic algorithm attempted to minimize the total error displayed by the system with respect to the provided data.

$$C(Positive(O), Negative(O)) = \begin{cases} Yes & ;\ Positive(O) > Negative(O) \\ No & ;\ Else \end{cases} \quad \textbf{(9)}$$

5 Experimental Results

5.1. Performance Evaluation of the Individual Classifiers

Each neural network performed perfectly on the training data set and displayed a recognition rate of 100% when presented with them after training. The test sequences were exercised through the four neural networks and the results obtained are presented in Fig. 3. The obtained results were then used to calculate the specificity, sensitivity and precision of the four neural networks.

As pointed out in section 2, the design of traditional promoter finding systems involves the coding of algorithms that would recognize the motifs present along a promoter strand. No consideration of these motif's was made during the design of this system. Nevertheless, it was observed that the weights of the neural networks automatically adjusted themselves in order to identify them. Fig. 4 is a graph representing the weights between the input layer and the first hidden layer within the network trained on the E1 encoding scheme. This diagram shows remarkable equivalence to Fig 1 in the fact that it lays emphasis[7] on the inputs at exactly the same positions as the motifs identified. Thus, manual encoding for the identification of these motifs was not necessary during the design process.

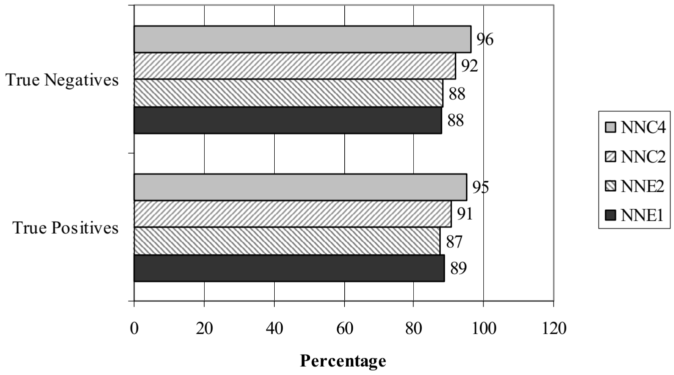


Fig. 3. The percentage of true positives recognized by the individual neural networks

[7] Larger weights

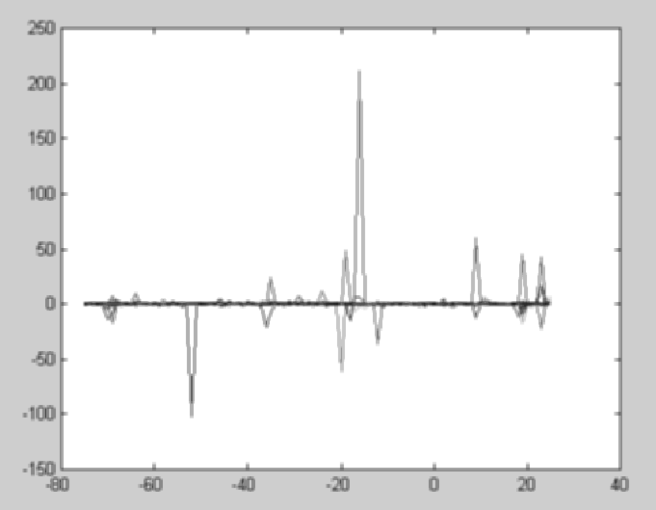


Fig 4. The magnitude of the weights between the input layer and the first hidden layer of the neural network trained on the E1 encoding method.

5.2 Performance Evaluation of the Combined System

The specificity, sensitivity and precision displayed by the combined system on the testing set using the LOP2 method are listed out in Tables 2 and 3. These results are compared, in Fig. 5, with the results obtained for the four individual neural networks. The results indicate that the combination of these four neural networks provides us with a far better recognition rate in terms of precision, specificity and sensitivity. Thus, the performance of the system has been improved by a considerable margin.

Table 2. The results obtained when the test datasets were exercised through the combined system.

	True	False
Positives	155	4
Negatives	169	2

Table 3. The Specificity, Sensitivity, Precision on the test datasets for the combined system

Attribute	Value

Specificity	0.9748
Sensitivity	0.9883

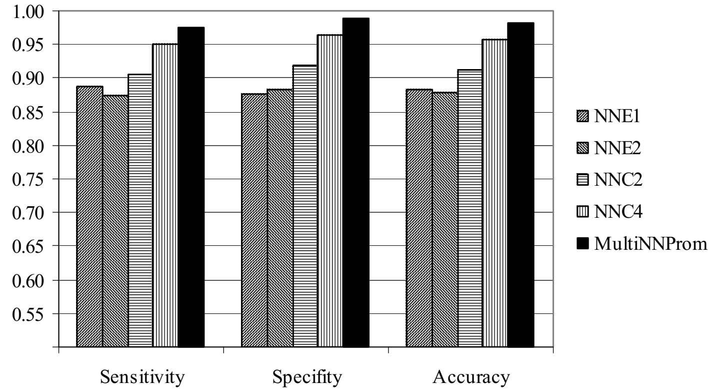


Fig. 5. Comparison of the specificity, sensitivity and precision of the four neural networks and the combined system

The outputs of the four neural networks were also combined using majority voting and the LAP method. Table 4 lists out a comparison on the specificity, sensitivity and precision between these two methods and the LOP2 method. These results point our the fact that, although the combination of the four neural networks using either method provides us with better results than the individual neural networks, the results obtained through the LOP2 method are better than those obtained while using the other two aggregating methods.

Table 5 compares the results obtained by us, using the approach described in this paper, with previous work done on the same problem. Thus, our system

Table 4. Comparison of the specificity, sensitivity and precision of the results obtained using different aggregating methods

Aggregating method	Specificity(S_p)	Sensitivity(S_n)	Precision(P)
Majority Voting	0.9560	0.9591	0.9515
LAP	0.9686	0.9708	0.9636
LOP2	0.9748	0.9883	0.9818

Table 5. Comparison of our results with previous work on the same problem

	S_p	S_n	P
MultiNNProm	0.9748	0.9883	0.9818
Ma et al., 2002	0.9176	0.9920	0.9194
Mahadevan and Ghosh, 1994	0.9020	0.9800	0.9040

is seen as a considerable improvement compared with recent research on the recognition of E.Coli promoters.

6 Conclusion

In this paper, we presented a novel approach for the recognition of E.Coli promoters in strings of DNA. The proposed system showed a substantial improvement in the recognition rate of these promoters, specifically in the recognition of true positives, i.e. rejection of non-promoters. This made our system display a far higher specificity than all other systems developed thus far.

We see the reason for this improvement to be twofold. First, it is the use of larger neural networks than those that have been used thus far. This led to a far better rate or recognition and generalization. We also note that an increase in the number of hidden layers within the neural network also led to a better recognition rate due increased abilities in classifying and identifying second and third order relationships between the individual base positions. Secondly, the use of multiple neural networks, each accepting the same set of inputs in different forms. We observed the fact that the false positive and false negative identifications by one network could be negated by true positive and true positive identifications by other networks. Thus, the combination of the opinions of many classifiers led us to a system that performed far better than the individual components.

References

[1] Baldi P., Brunak S, “Bioinformatics – The Machine Learning Approach”, MIT Press, Cambridge MA, 1998.

[2] Birney, E. "Hidden Markov Models in Biological Sequence Analysis". IBM Journal of Research and Development Volume 45, Numbers ¾, 2001.

[3] Hansen J.V., Krogh A., “A general method for combining in predictors tested on protein secondary structure prediction”, citeseer.ist.psu.edu/324992.html.

[4] Henderson, J., Salzberg, S. and Fasman, K. "Finding Genes in DNA with a Hidden Markov Model". Journal of Computational Biology, Vol. 4, No. 2 (1997), pp. 127-141.

[5] Koza J.R, Andre D., “Automatic Discovery of Protein Motifs Using Genetic Programming”, Evolutionary Computation: Theory and Applications, 1995.

[6] Krogh, A. "Two Methods for Improving Performance of a HMM and Their Application for Gene Finding". Proceedings of the Fifth International Conference on Intelligent Systems for Molecular Biology, AAAI Press, Menlo Park, CA, 1997, pp. 179-186.

[7] Kulp, D., Haussler, D., Reese, M. G. and Eeckman, F. H. Ä Generalized Hidden Markov Model for the Recognition of Human Genes in DNA". Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology, AAAI Press, Menlo Park, CA, 1996, pp. 134-142.

[8] Ma Q., Wang J.T.L., Wu C.H., “Application of Bayesian Neural Networks to Biological Data Mining: A Case Study in DNA Sequence Classification”, "citeseer.ist.psu.edu/314880.html".

[9] Mahadevan I. , Ghosh I. , “Analysis of E.Coli promoter structures using neural networks, Nucleic Acids Research, Vol 22, Issue 11 2158-2165, 1994.

[10] Ohno-Machado L., Vinterbo S., Webber G., “Classification of Gene Expression Data Using Fuzzy Logic”, Decision Systems Group.

[11] Riis S.K., Krogh A., “Improving prediction of protein secondary structure using neural networks and multiple sequence alignments”, Journal of Computational Biology, 3:163-183, 1996.

[12] Rogova G., “Combining the results of several neural network classifiers”, Neural Networks, 7(5):777-781, 1994.

[13] Rost B., Sander C., “Prediction of protein secondary structure at better than 70% accuracy”, Journal of Molecular Biology, 232(2):584-599, Jul 20, 1993.

[14] Salzberg S., Delcher A.L., Fasman K.H. , Henderson J., “A Decision Tree System for Finding Genes in DNA”, Journal of Computational Biology, 1997.

[15] Sharkey A.C.J., Sharkey N.E., “Combining diverse neural networks”, The Knowledge Engineering Review, 12(3):231-247, 1997.

[16] Snyder E.E. , Stormo G. , “Identification of Protein Coding Regions in Genomic DNA”, Journal of Molecular Biology(1995) 248, 1-18.

[17] Uberbacher E.C., Mural R. J., “Locating protein-coding regions in human DNA sequences by a multiple sensor-neural network approach”, Proc. Natl. Acad. Sci. USA, Vol 88, 11261-11265, 1991.

[18] Woolf P.J., Wang Y., “A Fuzzy Logic Approach to Analysing Gene Expression Data”, Physiol Genomics, 3: 9-15, 2000.

[19] Zenobi G., Cuningham P., “Using diversity in preparing ensembles of classifiers based on different feature subsets to minimize generalization error”, in proceedings of the 12th European Conference on Machine Learning, pages 576-587, 2001.

An Overview of Soft Computing Techniques Used in the Drug Discovery Process

Abiola Oduguwa, Ashutosh Tiwari, Rajkumar Roy and Conrad Bessant

Dept. of Enterprise Integration, Cranfield University,
Cranfield, Bedfordshire, UK. MK43 0AL.

Abstract. Drug discovery (DD) research has evolved to the point of critical dependence on computerized systems, databases and newer disciplines. Such disciplines include but are not limited to bioinformatics, chemoinformatics and soft computing. Their applications range from sequence analysis methods for finding biological targets to design of combinatorial libraries in lead compound optimisation. This paper presents a brief overview of classical techniques in DD with their limitations, and outlines current SC based techniques in this area.

Keywords: Drug discovery, genetic algorithm, neural networks, fuzzy logic, bioinformatics, chemoinformatics

1 Introduction

The discovery, development and approval of effective drugs for treatment and prevention of diseases in the pharmaceutical industry is a huge endeavor. Only a small fraction of the drug discovery (DD) projects undertaken eventually lead to successful medicines. Such programmes can take between 12 -16 years. Increasingly, the industry is compelled to find novel drugs that are more effective and safer than existing ones. New methods for DD are therefore receiving considerable attention in the industry. This is largely driven by several genome projects since the realization that many common human diseases have a genetic component. An example is the human genome project (HGP) which represents one of the major technical advances in genomics. It provided the first draft of the sequence of all genes in the human body which in turn aids understanding the genetic basis of diseases. The knowledge of the genetic base of disease helps gain deeper insights into the complex mechanisms in the human system and therefore, hints at possibilities to interfere with disease processes. The pharmaceutical industry has thus embraced the field of genomics as a source of novel drug target in their DD process. Developments in areas of proteomics, bioinformatics and chemoinformatics are also providing solutions to the need to enhance the DD process. Bioinformatics is used to exploit the data produced on this genome-wide scale. Proteomics studies help to understand the role of gene products (protein) particularly structural information of protein for use in drug design. Chemoinformatics supports drug research by creating tools for evaluating the molecular properties of potential drug compounds in the chemical databases.

Soft computing (SC) techniques are emerging as solution alternatives for dealing with the problems of biological sciences and DD research (Pedersen and Moult 1997; Manallack and Livingstone 1999; Chang and Halgamuge 2002). It is being applied in several areas of the DD research to achieve better solutions. SC is a collection of methodologies including as its main constituents Evolutionary Computation (EC), Fuzzy Logic (FL), Neuro-computing (NC), probabilistic computing (PC), chaotic theory and parts of machine learning theory. These methodologies are parallel to the remarkable ability of the human brain to reason and learn in an environment of uncertainty, imprecision, and implicit knowledge to achieve tractability, robustness and low cost solutions(Oduguwa 2003). SC differs from conventional techniques by providing an attractive opportunity to represent the ambiguity in human thinking with real life uncertainty which can result in more realistic solutions. It is these features that make SC a promising technology for dealing with DD problems.

This paper therefore focuses on the description of fuzzy logic, neural networks and evolutionary computing as they are the main SC technologies mostly used in drug and biomedical research. In this perspective, the principal contribution of fuzzy logic is to provide algorithms for dealing with imprecision and approximate reasoning. Neural networks provide an effective methodology for learning from examples and evolutionary computing provides a way to solve search and optimisation problems. Genetic algorithms, genetic programming, fuzzy logic and combinations of these techniques have been used to obtain better predictions in drug research (Langdon, Barret et al. 2002).

This paper is organised as follows. Section 2 gives an overview of the DD process including the classical techniques used in DD; section 3 introduces soft computing and the application areas in DD research. Section 4 discusses key findings and finally section 5 concludes this paper.

2 Drug Discovery

Drug discovery (DD) is a process of developing drugs for the safe and effective treatment of a disease. This process identifies, evaluates, and optimizes compounds and molecules with desired biological activity against a specified target or function (Verkman 2004). A target is a molecule within that plays a fundamental role at the onset or progression of a particular disease (Theullon-Sayag 2002). This section presents an overview of the DD process and the different stages of the process are described as follows.

2.1 The Drug Discovery Process

The DD process starts with a molecular target which originates from the discovery of a new gene or from the elucidation of the molecular mechanism of a genetic defect. Subsequent studies are performed to validate the target. Once suitability for DD has been established, new chemical entities are identified through random screening and/or rational drug design. The chemical leads with positive response in the screening process are selected and optimised as potential drug candidates. The result is a compound, or a small number of compounds that proceed to clinical trials for development.

The DD process can be divided into four main steps: drug target identification, target validation, lead compound identification and lead optimisation. This process is depicted in figure 1 and described briefly in the section that follows.

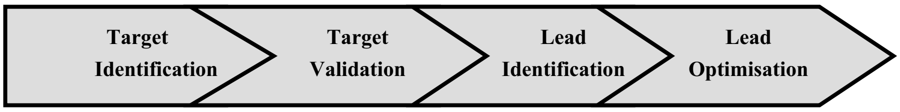


Figure 1: The drug discovery process

2.1.1 Target Identification

This stage of the DD process aims to identify genes or gene products that may be correlated with a disease process. This is achieved by quantifying and analyzing the gene expression in diseased and healthy states (Searls 2000; Knowles and Gromo 2002; Swindells and Overington 2002). The ultimate goal of this step is to find macromolecules that can become binding targets for potential drug compounds.

Target identification involves the isolation of the diseased tissue and analysis of its sequence using bioinformatics approaches such as sequence analysis and gene expression studies. Sequence alignment is an important tool in sequence analysis. It helps compare the structure-relationship between sequences by simultaneously aligning multiple sequences to construct connections between the elements in different sequences (Horng, Wu et al. 2004). It is used to identify shared regions of homology between a normal and a diseased gene sequence. The assumption is that the differential gene expression between the normal and diseased states indicates the gene involved in the disease and could therefore be a potential drug target.

Alignment of multiple sequences is characterised by very high computational complexity. Classical techniques used to solve this problem of sequence alignment include dynamic programming (Needleman and Wunsch 1970), three-dimensional matrix (Jue, Woodbury et al. 1980), progressive alignment, hidden markov models (HMMs), objective functions (Notredame and Higgins 1996), simulated annealing (Aart and Laarhoven 1987)and Gibbs sampling (Lawrence, Altschul et al. 1993). Dynamic programming was used to align two sequences on a 2-dimensional matrix. This was extended to compare three sequences on a 3-dimensional matrix. The progressive alignment is a simple and fast method that aligns sets of sequences with known tertiary structure. The disadvantage of this approach is the local minimum problem which stems from the nature of the algorithm. If any mistakes are made from any intermediate alignments, it cannot be corrected easily as more sequences are added to the alignment. HMM is an alternative approach used to simultaneously find an alignment and probability model of substitutions, insertions and deletions. This approach has a sound link in probability analysis but is limited in practice to cases with very many sequences (Notredame and Higgins 1996). The use of objective functions provides a measure of the quality of multiple alignments. An advantage of this approach is the confidence in the resulting alignment being the best by some criterion. However, the number of possible alignments which must be scored in order to choose the best one becomes astronomical for more than four or five sequences of reasonable length. Simulated annealing can be very slow and works well as an alignment improver. Gibbs sampling is very well in finding local multiple alignment block with no gaps but is not good at in gapped situations (Notredame and Higgins 1996; Horng, Wu et al. 2004).

2.1.2 Target Validation

Once the gene involved in a disease has been identified, it is necessary to validate them as drug targets. Target validation verifies the DNA or protein molecule that is directly involved in a disease process. That is, its role in disease must be clearly defined before drugs are sought that act against it. The aim is to understand the pathways/interaction of genes and to test whether the gene has the potential to be a therapeutic target (Searls 2000; Knowles and Gromo 2002; Swindells and Overington 2002), therefore target validation is a crucial step in the DD process.

Several methods are reported in the literature for validating genes as drug targets. They vary from hypothesis-driven studies of single genes in model organisms and in vitro cell-based assays to global scans for genes underlying disease processes (Chanda and Caldwell 2003; Hillisch and Hilgenfield 2003). The systematic alteration of single gene function in a relevant animal model represents a very powerful approach for drug target validation. However, its main disadvantage is the high cost and low speed of conventional animal models (small rodents). The *in vitro* assay technique relies typically on simple interactions of chemicals with a drug target. However, these cell-based assay results often poorly correlates with *in vivo* results because the complicated physiological environment is absent in the *in vitro* testing system (Ma 2004). Moreover, cell lines are usually transformed, exhibiting different gene expression and cell cycle profiles that those of cells in the living organism. In comparison, DNA micro array technology is an evolving high throughput method used in target validation. It permits simultaneous measurements of the expression levels of thousands of genes on a gene chip (Marton, DeRisi *et al.* 1998; Khan, Wei *et al.* 2001). The gene expression data from the gene chip is often organized as an $n \times m$ matrix $\mathbf{M}$ whose rows are the n genes and whose columns are the m biological samples in which their expression has been monitored. Several statistical and computational methods have been developed to identify and select key predictive genes for use in classification. Examples include cluster analysis and principal component analysis. Currently most of these reports have focused on situations where the expression dataset being analysed contains only two to three major classes for example, cancer versus normal tissue, response to drug versus no response (Ooi and Tan 2003). However, when the dataset contains multiple classes (greater than five), the methodologies that suffice for two to three-class datasets may not necessarily produce comparable accuracies in the larger, more complex datasets. This therefore indicates a strong need for the development of better algorithms that can effectively analyse multiple-class expression data.

2.1.3 Lead Identification

This is the process of identifying biologically active chemical entities that could be optimized into drugs. In this stage, compounds which interact with the target protein and modulate its activity are identified. The methods used for identifying the compounds are random screening or directed design approach and virtual screening approach (Castrodale 2002; Hillisch and Hilgenfield 2003).

High-throughput Screening (HTS) is used to test large collections of compounds or structurally selected compounds in the database for their ability to affect the activity of the target protein. The compound database screened is made up of millions of compounds. These compounds are screened with a throughput of 10 000 (HTS) up to 100 000 compounds per day (μHTS, ultra high throughput screening) (FitzGerald 2000; Bleicher, Bohm et al. 2003; Hillisch and Hilgenfield 2003; Verkman 2004). The main problem of HTS is cost. It can cost up to ~$300K to set up a high throughput screen. And for most companies HTS is not an option because it can take weeks or months to scale a low throughput screen up to high throughput capacity. Additionally, many screens are not yet possible with high throughput techniques. Establishing a robust assay for a new target takes time and data quality of high throughput screens is often problematic. Hit rates against some receptors are reported to be very low, necessitating screening of very large numbers of compounds (tens to hundreds of thousands) (Bleicher, Bohm et al. 2003).

The second approach to lead identification is *in silico* or virtual screening. It involves the use of computational analysis to select a subset of compounds considered to be appropriate for a given receptor. Three-dimensional structures of compounds from virtual or physically existing libraries are docked into the binding sites of target proteins with known or predicted structure. Empirical scoring functions are used to evaluate the fit between the compounds and the target protein. The highest ranked compounds are suggested for further biological testing. One of the problems of virtual screening is the availability of protein structures. Structure prediction methods include computationally intensive strategies that simulate the physical and chemical forces involved in protein folding, and knowledge–based approaches that use information from structure databases to build models. Despite several years of research, structure prediction problem is still unsolved (Clark and Pickett 2000; Bajorath 2001; Maggio and Ramnarayan 2001). Experimental determination of protein structures by X- ray crystallography is time consuming and expensive (Hillisch and Hilgenfield 2003); (Maggio and Ramnarayan 2001). Current prediction techniques like homology modeling and threading techniques require at least an experimentally determined protein in a fold class to model hundreds of related proteins. These techniques involve using database search tools to identify similarity between sequences and structures. The researcher then identifies the biological significance sequences and determines if the sequences are derived from a common ancestor. These rational techniques however, produce low resolution models due to lack of adequate understanding on how the primary structure of the protein determines its tertiary structure.

2.1.4 Lead Optimisation

Lead optimisation is the complex multi-step process of refining the chemical structure of a hit to improve its drug characteristics, with the goal of producing a pre-clinical drug candidate. This process generally involves iterative rounds of synthetic organic chemistry and compound evaluation of a potential drug compound to ensure optimal properties in drug development (Hillisch and Hilgenfield 2003; Verkman 2004). These properties include potency, adsorption, metabolism, distribution, toxicity (ADME/Tox). This process is time-consuming (Stahura and Bajorath 2002; Bleicher, Bohm et al. 2003; Hillisch and Hilgenfield 2003). Typical process time ranges from months to years.

The process of lead optimisation begins with evaluating hits in secondary test assays and analogs (a set of related compounds) which are then synthesized and screened. The resulting quantitative information is known as structure-activity relationships (SAR). SAR shows the relationship between chemical structure and biological data. Verkman *et al* (Verkman 2004) reported several approaches that are available to maximise the utility of this SAR information for directed acquisition and synthesis of structural analogues to improve compound potency. In terms of utilizing SAR data to accelerate compound optimisation, visual inspection reveals many important structural features associated with activity. Several computational approaches are also reported in the literature adopted in lead optimisation; these are rational drug design, pharmacophore analysis, and quantitative structural activity relationship (QSAR) analysis. These approaches are used individually or combined in various forms. The rational design method involves the use of a high resolution structure of a target to direct the synthesis of new analogues. The process usually involves generation of a large library of potential derivatives and use of computational docking methods to select derivatives that may interact with the target on the basis of shape complementarities or charge placement. The pharmacophore methods involve definition of a minimal unit (e.g. hydrophobic group or other functional groups) that leads to activity in 3 dimensional space. The consensus pharmacophore is then used to examine the allowed placements of groups in a set of candidate compounds. The last method is to establish QSAR models. These relate calculated physiochemical properties of molecules to activity, rather than strictly structural characteristics. It usually requires a set of structurally related compounds with a wide range of activities. The main limitation of these techniques is that they are labour-intensive and time consuming. Additionally, there is a need to have a better understanding of the mechanism underlying toxicity of drugs. Current methods using animal models are time-consuming, low throughput and can be unreliable.

2.2 Limitations of Classical Techniques

The techniques discussed have proven useful in DD research; however the limitations of these techniques need to be addressed. These limitations include visual methods applied to gene expression data to identify gene patterns; such solution is not always generalisable. Also, insufficient availability of structural information hinders the process of rational drug design, the use of cell-based assays to predict efficacy and toxicity of hits also poses problem of reproducibility. This shows the need for more robust methods to address these problems and yield better solutions. A summary of these problems/limitations is presented in table 1.

Table 1: Summary of Classical Techniques

DD Process	Classical Techniques	Problems
Target Identification	• Sequence analysis • Sequence alignment • DNA Microarray	• Labour intensive • Visual inspection of data is subjective Long computational time
Target Validation	• Animal models • DNA Microarray	• Labour intensive • Time Consuming • Highly dimensional and complex data Insufficient understanding of genetic pathways
Lead Identification	• HTS • Virtual screening	• Expensive • Not applicable to all compounds • Availability of 3D structures of gene product Production of low resolution models
Lead Optimisation	• In vivo method • Rational drug design • QSAR studies	• Labour intensive • Time consuming • Expensive • Reproducibility • Insufficient knowledge

3 Soft Computing Techniques in Drug Discovery

SC is now emerging as a solution alternative to deal with some of the issues outlined in the previous section. The past few years have witnessed phenomenal growth in the application of SC techniques in DD research with its increasing use in several areas biological sciences research. SC is an emerging research area in DD. This research area attempts to exploit SC to deal with imprecise information and provide novel solutions to DD problems.

This section presents an overview of interesting applications of SC techniques in the target identification, target validation, lead identification and lead optimisation stages of the DD process as described in section 2. Table 1 presents a summary of the proportion of different SC techniques in the sample of publications reviewed in this paper.

3.1 SC in Target Identification

Several authors have developed sequence alignment algorithms based on SC techniques (Reijmers, Wehrens et al. 1999; Deutsch 2003; Ooi and Tan 2003; Horng, Wu et al. 2004). Isokawa et al (Isokawa, Wayama et al. 1996) and Hanaya et al (Hanada, Yokoyama et al. 2000) applied simple GA with bit matrices to the problem of multiple sequence alignment. Notredame et al (Notredame and Higgins 1996) developed a software package, SAGA based on GA for aligning two homologous sequences of RNA sequence. GA has also been used for the construction of phylogenetic trees in sequence analysis studies (Reijmers, Wehrens et al. 1999). Another approach for identifying potential therapeutic targets is by using quantitative gene expression profiling. Deutch (Deutsch 2003) described an evolutionary algorithm identifying sub-sets of expressed sequences in gene expression data. Ooi and Tan (Ooi and Tan 2003) applied GA in a multi-class prediction problem to predict gene groups and Khan et al (Khan, Wei et al. 2001) developed a method of classifying tumor genes to specific diagnostic categories based on their gene expression profiles using artificial neural networks (ANNs).

3.2 SC in Target Validation

SC techniques have been developed to classify expression signatures of genes in a micro array analysis. Khan et al demonstrated the application of ANNs to validate SRBCTs (small round blue cell tumors) genes involvement in cancer development. Keedwell et al also described the use of neural–genetic hybrid algorithm for gene expression analysis (Keedwell and Narayanan 2003). In the study, GA was used to identify the possible gene combinations for classification and the output from the neural network was used to determine their fitness.

3.3 SC in Lead Identification

Literature reveals SC techniques have been applied in the field of rational drug design to generate new leads. Lead compounds are found in existing chemical databases by fast searching or docking protocols, synthesized and isolated by combinatorial chemistry, or designed *de novo* by computational design programs (Parrill 1996). SC techniques have been applied in each of these areas.

GA has been applied to the problem of finding two dimensional matches to a query in chemical databases (Brown, Jones et al. 1994; Clark, Jones et al. 1994; Parrill 1996). It has also been applied to the problem of comparing three-dimensional structures, both to determine optimal alignments of molecular electrostatic potential fields in rigid searches (Fontain 1992) and in flexible searching for a pharmacophoric pattern (Wild and Willett 1996).

Another method for lead identification is docking of ligands into the active site of a biological target (Glen and Payne 1995; Parrill 1996; Jagla and Schuchhardt 2000; Schneider, Clement-Chomienne et al. 2000; Schneider, Lee et al. 2000; Wang, Gao et al. 2000; Budin, Ahmed et al. 2001; Budin, Majeux et al. 2001; Pegg, Haresco et al. 2001). Computational methods for docking problems involve a good scoring function and an efficient searching algorithm for searching conformational spaces. Trial ligands are taken from a 3D database, placed into the template site, and ranked in order of predicted binding affinity. A variety of methods have been used to obtain plausible binding orientations. One philosophy requires a user to manipulate the ligand, while the computer interactively reports a binding score (Parrill 1996). Several groups have used GAs in this area (Jones, Willett et al. 1995; Oshiro, Kuntz et al. 1995; Bamborough and Cohen 1996; Parrill 1996; Desjarlais and Clarke 1998; Globus, J et al. 1999; Yang and Chen 2004). (Yang and Chen 2004) developed a genetic evolutionary method, GEMDOCK for molecular docking and empirical scoring function. The program combines discrete and continuous global search strategies to speed up convergence. Oshiro et al uses a force field scoring function to dock inhibitors to thymidylate synthase and to the two old favorites HIV protease and DHFR. A DOCK program (Oshiro, Kuntz et al. 1995; Parrill 1996) uses GA to dock flexible ligands into aigid receptor after characterizing and identifying binding sites using a surface sphere cluster method. For each of the iteration, selective pressure is applied to encourage high-scoring features of current generation to be preserved in the next cycle. Random mutations are permitted, while crossover moves allow molecules to exchange subsets of their characteristics. The method does not necessarily converge to give the same answer every time, so several runs may be needed.

In addition to finding lead compounds by the previously described methods, they can also be developed experimentally by *de novo* design (Glen and Payne 1995; Parrill 1996; Schneider, Clement-Chomienne et al. 2000; Budin, Ahmed et al. 2001; Budin, Majeux et al. 2001; Pegg, Haresco et al. 2001). Glen and Payne have applied GAs to the design of substructures based on a wide variety of user-defined constraints. The constraints selected for various design experiments provide the basis for the fitness function.

Other application areas of SC application in lead identification are protein folding simulations for predicting the three-dimensional structure of a target protein (Pedersen and Moult 1996; Pedersen and Moult 1997; Konig and Dandekar 1999; Agostini and Morosetti 2003; Cooper, Corne et al. 2003) and in combinatorial library design (Illgen, Enderle et al. 2000).

3.4 SC in Lead Optimisation

The application areas of SC in lead optimisation include: combinatorial chemistry, *de novo* design leads and quantitative structure activity relationship measurement (QSAR) (Parrill 1996).

Combinatorial chemistry involves the synthesis and screening of large libraries of compounds to determine lead compounds that exhibit biological activities of interest. GA has been applied in this area with successes in the design and automated synthesis of combinatorial compound libraries (Parrill 1996). The optimisation behavior of GA perfectly matches the discontinuous, non-steady structure space of chemistry. Lutz Weber demonstrated an application of GA for the development of thrombin inhibitors using non-peptidic molecules (Weber, Wallbaum et al. 1995; Weber 1998; Weber 1998). The study involves a Ugi reaction and 10x40x10x40 building blocks gives 160,000 combinatorial products. Another application of GAs in combinatorial chemistry is the selection of fragments for assembly into the library. Sheridan and Keasley (Weber, Wallbaum et al. 1995; Parrill 1996; Weber 1998; Weber 1998) applied GAs to the optimisation of tripeptoids constructed from a wide variety of primary and secondary amines. Each tripeptoid generated in the GA run was evaluated based on its similarity to a target tripeptoid. The ones with high fitness values were chosen for use during library synthesis. This method was useful for the development of specific biological screens.

Table 2: Summary of applications of SC in drug discovery

DD Process	Applications	Characteristics	SC component
Target Identification	Sequence Analysis	• A real coded GA for multiple sequence alignment • Determination of phylogenetic relationship between sequences using simple GA • Extraction of protein signatures with ANN and fuzzy logic	GA, FL, ANNs
	Gene Expression studies	• Classification of gene expression in a multiple class data • Classification of cancer gene using neural networks	GA, ANNs
Target Validation	Gene Expression studies	• Neural-GA hybrid algorithm for analysis of gene expression profiles Leukemia gene dataset • Classification of disease gene using neural networks	GA, ANNs
Lead Identification	Database Searching	• Comparison of two and three dimensional structures using GA	GA
	Protein Folding / Structure Prediction	• Coded GA for simulating protein folding problem •	ANNs, FL, GA
	Virtual Screening and molecular Docking	• Docking of ligands in to receptor sites of target protein molecule using GA	GA
	De novo drug design	• Design of protein substructures, receptors, enzymes, ion channels using GA	GA
Lead Optimisation	Combinatorial Chemistry	• Development of thrombin inhibitors with non-peptidic molecules • Design of similar and dissimilar compounds in a combinatorial library	GA
	De novo drug design	• Optimisation of leads generated by PRO-LIGAND program	GA
	QSAR	• BRANNs for producing better descriptors and designing molecules with desired activity	GA, ANNs

The application of SC in *de novo* design is shown in the optimisation of the results of the design program PRO_LIGAND (Parrill 1996). This program generates new lead compounds by assembling fragments for substructure libraries.GA uses these leads as the initial population for the optimisation process.

The third approach for optimisation of lead compounds is QSAR modeling. The method attempts to find relationship between the properties of bioactive molecules and the biological responses they produce when applied to a biological system. Winkler and Burden developed QSAR models to improve the efficiency of bioactive molecules with Bayesian regularized artificial neural networks (BRANNs) (Winkkler and Burden 2004). A hybrid GA/neural network method has also been used to suggest more potent descriptors from a set of variables in a QSAR model for dihydrofolate reductase inhibitors (Manallack and Livingstone 1999).

4 Discussion

This paper provided a brief overview of the classical techniques in DD. It was shown that these techniques have several limitations which led to the introduction of soft computing in DD primarily to increase efficiency.

This paper reveals the several application areas of soft computing techniques in the DD process. The main SC techniques are GA, FL, and ANNs. Literature reveals a number of studies of the application of GAs in various stages of DD over the last 15 years. This is largely due to the nature of the data and the search space being explored. Gene expression studies for target identification and validation is an example of where the GA approach has been successful. The attributes of gene expression data which makes GAs suitable is that values of the data are largely continuous, and most datasets consist of a huge number of genes (variables) and a small number of records. This is because tens of thousands of genes are sampled and only a small number of subjects are used. Also, many phases of the drug design involve finding solutions to large combinatorial problems for which exhaustive search is intractable. GA has been particularly useful in this area to rapidly find good solutions to such problems (Parrill 1996). ANNs have also been successful in gene expression data analysis and pattern recognition in protein sequence data. FL has also been shown to help in extraction of information from biological datasets. The key feature of DNA that makes it appropriate for fuzzy computing is the uncertainty and incompleteness in the information of the double stranded duplex. These three core techniques of soft computing have also been used in its hybrid form. For example, neuro-fuzzy algorithm that was used in protein motif extraction (Chang and Halgamuge 2002). The authors also applied this hybrid algorithm to improve the speed and flexibility of finding important signature in protein sequence compared to the other techniques. This paper has therefore shown the growing trend in SC based techniques notably GA, ANNs and FL in handling the imprecision and uncertainty in biological data.

5 Conclusions

This paper presented a review of the application of Soft Computing techniques in DD. It explores the discovery phase of drug research. The paper shows a brief review of the classical techniques in DD research with their limitations and critically evaluates how current SC techniques are suited for such complex biological science problems. Current research shows that SC methods have significant potential in dealing with the limitation posed by traditional methods.

Acknowledgments

The authors would like to thank Cranfield University, UK for financially supporting this research.

References

Aart, E. and V. P. Laarhoven (1987). Simulated Annealing: a Review of Theory and Applications. Amsterdam, Kluwer Academic Publishers.

Agostini, L. and S. Morosetti (2003). "A simple procedure to weight empirical potentials in a fitness function so as to optimise its performance in ab initio protein-folding problem." Biophysical Chemistry **105**: 105 - 118.

Bajorath, J. (2001). "Rational drug discovery revisited: interfacing experimental programs with bio- and chemo-informatics." Drug Discovery Today **6**(19): 989-995.

Bamborough, P. and F. E. Cohen (1996). "Modelling Protein-Ligand complexes." Current Opinion in Structural Biology **6**: 236 - 241.

Bleicher, K. H., H. Bohm, et al. (2003). "Hit and Lead Generation: Beyond High-Throughput Screening." Nature Review Drug Discovery **2**(5): 369-378.

Brown, R. D., G. Jones, et al. (1994). "Matching two-dimensional chemical graphs using genetic algorithms." Journal of Chemical Information and Computer Sciences **34**(1): 63 - 67.

Budin, N., S. Ahmed, et al. (2001). "An Evolutionary Approach for Structure-based Design of Natural and Non-natural Peptidic Ligands." Combinatorial Chemistry and HTS **4**: 661-673.

Budin, N., N. Majeux, et al. (2001). "Structure-based Ligand Design by a Build-up Approach and Genetic Algorithm Search in Conformational Space." Journal of Computational Chemistry(22): 1956-1970.

Castrodale, B. (2002). Leading Genomic Approaches for Breaking Bottlenecks in Drug Discovery and Development. Massachusetts, Cambridge Healthtech Institute**:** 1-7.

Chanda, S. K. and J. S. Caldwell (2003). "Fulfilling the promise: drug discovery in the post-genomic era." Drug Discovery Today **8**(4): 168-174.

Chang, B. C. H. and S. K. Halgamuge (2002). "Protein Motif Extraction with Neuro-Fuzzy Optimization." Bioinformatics **18**(8): 1084-1090.

Clark, D. E., G. Jones, et al. (1994). "Pharmacophoric pattern matching in files of three-dimensional chemical structures: Comparison of conformational-searching algorithms for flexible searching." Journal of Chemical Information and Computer Sciences **34**(1): 197-206.

Clark, D. E. and S. D. Pickett (2000). "Computational methods for the prediction of 'drug-likeness'." Drug Discovery Today **5**(2): 49-57.

Cooper, L. R., D. W. Corne, et al. (2003). "Use of a novel Hill-Climbing genetic algoriothm in protein folding simulations." Computational Biology and Chemistry **27**: 575 - 580.

Desjarlais, J. R. and N. D. Clarke (1998). "Computer search algorithms in protein modification and design." Current opinion in Structural Biology **8**: 471 - 475.

Deutsch, J. M. (2003). "Evolutionary Algorithms for Finding Optimal Gene Sets in Microaray Prediction." Bioinformatics **19**(1): 45-52.

FitzGerald, K. (2000). "*In vitro* display technologies - new tools for drug discovery". DDT. **5**(6): 253-258.

Fontain, E. (1992). "Application of genetic algorithms in the field of constitutional similarity." Journal of Chemical Information and Computer Sciences **32**(1): 748-752.

Glen, R. C. and A. W. R. Payne (1995). "A genetic algorithm for the automated generation of molecules within constraints." Journal of Computer-Aided Molecular Design. **9**(2): 181-202.

Globus, A., L. J, et al. (1999). "Automatic molecular design using evolutionary techniques." Nanotechnology **10**: 290 - 299.

Hanada, K., T. Yokoyama, et al. (2000). Multiple Sequence Alignment by Genetic Algorithm. Genome Informatics. **11:** 317 -318.

Hillisch, A. and R. Hilgenfield (2003). Modern Methods of Drug Discovery, Springer Verlag.

Horng, J., L. Wu, et al. (2004). "A genetic algorithm for mutiple sequnce alignment." Soft Computing.

Illgen, K., T. Enderle, et al. (2000). "Simulated molecular evolution in a full combinatorial library." Chemisry and Biology **7**: 433- 441.

Isokawa, M., M. Wayama, et al. (1996). "Multiple Sequence Alignment Using Genetic Algorithm." Genome Informatics **7**: 176 - 177.

Jagla, B. and J. Schuchhardt (2000). "Adaptive Encoding Neural Networks for the Recognition of Human Signal Peptide Cleavage Sites." Bioinformatics **16**: 245-250.

Jones, G., P. Willett, et al. (1995). "Molecular recognition of receptor sites using a genetic algorithm with a description of desolvation." Journal of Molecular biology **245**: 43-53.

Jue, R. A., N. W. Woodbury, et al. (1980). "Sequence homologies among e. coli ribosomal proteins: evidence for evolutionary related groupings and internal duplications." Journal of Molecular Evolution **15**: 129-148.

Keedwell, E. C. and A. Narayanan (2003). Genetic algorithms for gene expression analysis. Applications of Evolutionary Computation: Proceedings of the 1st European Workshop on Evolutionary Bioinformatics(EvoBIO 2003), Springer Verlag LNCS.

Khan, J., J. S. Wei, et al. (2001). "Classification and diagnostic prediction of cancers using gene expression profilling and artificial neuarl networks." nature Medicine **7**(6): 673-679.

Knowles, J. and G. Gromo (2002). "Target Selection in Drug Discovery." Nature Reviews: Drug Discovery **2**: 63-69.

Konig, R. and T. Dandekar (1999). "Improving genetic algorithms for protien folding simulations by systematic crossover." BioSystems **50**: 17 - 25.

Langdon, W. B., S. J. Barret, et al., Eds. (2002). Genetic Programming for combining neural networks for drug discovery. Soft Computing and Industrial Application, Springer-Verlag.

Lawrence, C., S. Altschul, et al. (1993). "Detecting subtle sequence signals: a gibbs sampling strategy for multiple alignment." Science **262**: 208-214.

Ma, C. (2004). Animal models of diseases. Mdern Drug Discovery. **3:** 30 - 36.

Maggio, E. T. and K. Ramnarayan (2001). "Recent developments in computational proteomics." Drug Discovery Today **6**(19): 996-1004.

Manallack, D. T. and D. J. Livingstone (1999). "Neural networks in drug discovery: have they lived up to their promise?" Europen Jopurnal of Medicinal Chemistry **34**: 195 - 208.

Marton, M. J., J. L. DeRisi, et al. (1998). "Drug target validation and identification of secondary drug target effects using DNA microarrays." Nature Medicine **4**(11): 1293 - 1301.

Needleman, S. B. and C. D. Wunsch (1970). "A general method applicable to the search for similarities in the amino acid sequences of two proteins." Journal of Molecular Biology **42**: 245-261.

Notredame, C. and D. G. Higgins (1996). "SAGA: sequence alignment by genetic algorithm." Nucleic Acids Research **24**(8): 1515-1524.

Oduguwa, V. (2003). Rolling System Design Optimisation Using Soft Computing Techniuques. Enterprise Integration. Bedfordshire, Cranfield**:** 332.

Ooi, C. H. and P. Tan (2003). "Genetic Algorithms Applied to Multi-Class prediction for the Analysis of Gene Expression Data." Bioinformatics **19**(1): 37-44.

Oshiro, C. M., I. D. Kuntz, et al. (1995). "Flexible ligand docking using a genetic algorithm." Journal of Computer-Aided Molecular Design **9**(1): 113-130.

Parrill, A. (1996). "Evolutionary and genetic methods in drug design." DDT **1**(12): 514 - 521.

Pedersen, J. T. and J. Moult (1996). "Genetic algorithms for protein structure prediction." Current Opinion in Structural Biology **6**: 227 -231.

Pedersen, J. T. and J. Moult (1997). "Protein Folding simulations with genetic algorithms and a detailed description." Journal of Molecular Biology **269**: 240 - 259.

Pegg, S. C. H., J. J. Haresco, et al. (2001). "A Genetic Algorithm for Structure-based De Novo Design." Journal of Computer-Aided Molecular Design. **15**: 911-933.

Reijmers, T. H., R. Wehrens, et al. (1999). "Quality Criteria of Genetic Algorithm for Construction of Phylogenetic Trees." Journal of Computational Chemistry **20**(8): 867 - 876.

Schneider, G., O. Clement-Chomienne, et al. (2000). "Virtual Screening for Bioactive Molecules by Evolutionary De Novo Design." Angewandte Chemie International Edition in English **39**: 4130-4133.

Schneider, G., M.-L. Lee, et al. (2000). "De novo design of molecular architectures by evolutionary assembly of drug-derived building blocks." Journal Computer-Aided Molecular Design **14**: 487-494.

Searls, D. B. (2000). "Using Bioinformatics in gene and drug discovery." DDT. **5**(4): 135-143.

Stahura, F. L. and J. Bajorath (2002). "Bio- and chemo-informtics beyond data management: crucial challenges and future opportunities." Drug Discovery Today **7**(11): s41-s47.

Swindells, M. B. and J. P. Overington (2002). "Prioritizing the proteome: identifying pharmaceutically relevant targets." Drug Discovery Today **7**(9): 516-521.

Theullon-Sayag, V. (2002). Impact of e-Pharma Technology on the Drug Development Process. Enterprise Integration. Cranfield, Cranfield University**:** 93.

Verkman, A. S. (2004). "Drug Discovery in academia." American Journal of Physiology - Cell Physiology **286**: C465-C474.

Wang, R., Y. Gao, et al. (2000). "LigBuilder: A Multi-Purpose Program for Structure-based Drug Design." Journal of Molecular Modeling **6**: 498-516.

Weber, L. (1998). "Application of genetic algorithms in molecular diversity." Currnet Opinion in Chemical Biology **2**: 381 - 385.

Weber, L. (1998). "Evolutionary combinatorial chemistry: application of genetic algoithms." Drug discovery Today **3**(8): 379 - 385.

Weber, L., S. Wallbaum, et al. (1995). "Optimisation of the Biological Activity of Combinatorial Compound Libraries by a Genetic Algorithm." Angewandte Chemie International Edition in English **34**: 2280-2282.

Wild, D. J. and P. Willett (1996). "Similarity Searching in Files of Three-Dimensional Chemical Structures. Alignment of Molecular Electrostatic Potential Fields with a Genetic Algorithm." Journal of Chemical Information and Computer Sciences **36**(2): 159-167.

Winkkler, D. A. and F. R. Burden (2004). "Bayesian neural nets for modeling in drug discovery." DDT: BIOSILICO **2**(3): 104-111.

Yang, J.-M. and C.-C. Chen (2004). "GEMDOCK: A generic evolutionary method for molecular docking." PROTEINS: Structure, Function, and Bioinformatics **55**: 288 - 304.

Part XII

Text Processing

Ontology-Based Automatic Classification of Web Pages

Mu-Hee Song Soo-Yeon Lim Seong-Bae Park Dong-Jin Kang* Sang-Jo Lee

{Dept. of Computer Engineering, Information Technology Services*} Kyungpook National University, Daegu, Korea
mhsong@mail.knu.ac.kr nadalsy@hotmail.com {seongbae, djkang, sjlee}@mail.knu.ac.kr

Abstract. The use of ontology in order to provide a mechanism to enable machine reasoning has continuously increased during the last few years.This paper suggests an automated method for document classification using an ontology, which expresses terminology information and vocabulary contained in Web documents by way of a hierarchical structure. Ontology-based document classification involves determining document features that represent the Web documents most accurately, and classifying them into the most appropriate categories after analyzing their contents by using at least two pre-defined categories per given document features. In this paper, Web documents are classified in real time not with experimental data or a learning process, but by similar calculations between the terminology information extracted from Web texts and ontology categories. This results in a more accurate document classification since the meanings and relationships unique to each document are determined.

Keywords: Document classification, Ontology, Web Page classification, term weighting scheme

1. Introduction

As the Internet usage rate has rapidly increased, the volume of electronic documents that can be seen on the Web has also substantially increased. Since so many of these electronic texts now exist in terms of their pure numbers and volumes, it is impossible for people to individually classify information contained in millions and millions of Web pages. Thus, there is an increasing need for a tool, which can classify electronic documents into appropriately pre-defined categories.

As a means to resolve the problem of classifying an almost infinite volume of electronic documents accurately and efficiently, an automated document classification method, using the ontology method, is suggested in this paper. Typically, Web documents have the following characteristics: First, they are structured in a Web site with the Web site as a unit, in other words, a Web site normally consists of many Web texts with the same or a similar subject. Second, each Web document exists as a part of other Web sites. Finally, individuals or organizations that have specific purposes manage typical Web sites.

This paper considers the fact that terminology information can be easily extracted using such Web text characteristics. In our research, Web documents are classified based on similarities determined by the ontology, which expresses the meaning structure of the Web documents' terminology information and vocabulary in a hierarchical manner. Identifying and comparing the meaning content and relationship of each document can perform document classification more accurately and efficiently. The ontology mentioned in this paper is comprised of concepts, concepts features, relations between concepts, and constraints for document classification, all in a hierarchical manner. Also, the ontology's hierarchical structure is applied to the document classification.

Our work is distinguished from others for the following reasons: (1) Rather than using a dictionary or knowledge index, ontology is used for document classification. (2) Our ontology is based on syntax information contained in the Web pages. (3) Mapping between the established ontology and terminology information extracted from Web pages is performed.

The second section of this paper provides background information about document classification and ontology through discussions on related research. The third section details how ontology, as suggested in this paper, is applied to document classification. The experiments and evaluation of web page classification in section fourth and the final section outlines the conclusion and future research.

2. Related works and Background information

Research work on document classification systems for efficiently managing and searching large amounts of electronic text has been ongoing for quite some time. This section discusses the existing research work, and what the ontology theory is all about.

2.1 Document Classification

Document classification involves determining the document features that represent documents most appropriately, and placing the documents into the most related categories after evaluating its contents, by using at least two pre-defined categories based on the already defined document features. This section describes research which has been carried out by people in the area of automatic document classification, and it examines the key difference between our work and other research.

The rule-based model [1] utilizes experts' help based on generally distinguished rules that appear in study texts or applies rules that are extracted by studying the documents. The Bayesian probability model [2,3] applies the probability theory to the document features extracted from the documents. The SVM (Support Vector Machine: SVM) [4] uses the machine learning method. The text which is to be classified from the information search standpoint, is queried and similar texts are searched in the *k*-nearest neighbor model [5]. Although these methods have some degree of accuracy, all of them require some rule learning level and they must have the training data as a reference.

This paper suggests an ontology-based automated document classification method, which does not require learning processes and can be performed in real time.

Our work is different from other research in that we focus on (1) the use of domain ontologies, rather than a dictionary or thesaurus, to assist classification; (2) building ontologies based on text information within the classification schemes and Web pages; (3) the mapping between a set of ontologies and the classification schemes.

2.2 Ontology

Ontology applications have recently become important factors in building the Semantic Web, whereby information is given explicit meaning, making it easier for machines to automatically process and integrate the data information available on the Web. Research on Web-based ontology has been expanding in terms of its potential applications. While ontology can be seen as a database in a large sense., it is more related to knowledge which has a more complex form than data, and it is also referred to as a knowledge base. In the strictest sense, however, ontology is just another form of a database. This is different from a knowledge base, because it deals more with conceptual relationships amongst terms and vocabulary than knowledge itself, which has a greater comprehensive meaning and includes both the knowledge content and logical reasoning processes. In other words, ontology is a conceptual expression of common terms used in written documents with specific subjects [6].

3. Document Classification using Ontology

3.1 Ontology Structure

In this paper, ontology is defined as one independent, collective representation of all standardized concepts for vocabulary and terms in one place. Here, we are not talking about collections of simple words, but we are referring to collection of vocabulary, which have relationships with both simple rules and meanings. Ontology expressions are based not only on the logical relations between term definitions and other meanings, but also on the bottom-out structure where the interpretation starts from primitive terms. We have decided to apply ontology to Web document classification, because it has the unique, hierarchical structure and characteristic of machine reasoning, starting from very primitive terms.

The advantages of an ontology-based classification approach over the existing ones, such as hierarchical, - [7] and probabilistic – approach [8], are that (1) the nature of the relational structure of an ontology provides a mechanism to enable machine reasoning; (2) the conceptual instances within an ontology are not only a bag of keywords but have inherent semantics, and a close relationship with the class representatives of the classification schemes. Hence, they can be mapped to each other; (3) this is a kind of Web page and class representative. It also enables us to get insights into and observe the way the classifier assigns a class representative to a Web pages by tracking the links between the conceptual instances involved and the associated class representative [9].

The main factors and relationships in this paper's ontology are listed below.

- Main Factors in Ontology
  - Concept: Generally used terms in specific domains
  - Feature: Used to describe an individual item
  - Relation: Defines and connects relationship patterns among concepts
  - Constraint: Required condition to promote each individual item

- Relationships in Ontology
  - Equal (E-R): When meanings between concepts are the same.
  - Succession (is-A): When the class and the instance are the same
  - Part (has-A): When A has B as part of A (has-part, part-of)

3.2 Building Domain Ontology for Document Classification

In this study, ontology for the 'economy' domain has been developed for experimental purposes of document classification. To configure and develop the ontology, it is first assumed that vocabulary which frequently appears in document collection is similarly related to other vocabulary. The second point is this frequently appearing vocabulary is used to build the basic network structure. Third, adding vocabulary that has a relationship with those selected words expands the ontology. Then, similarities between the terminology information extracted from Web page, and ontology terminology data are identified and compared in order to start the document classification process. Figure 1 shows a portion of the domain ontology.

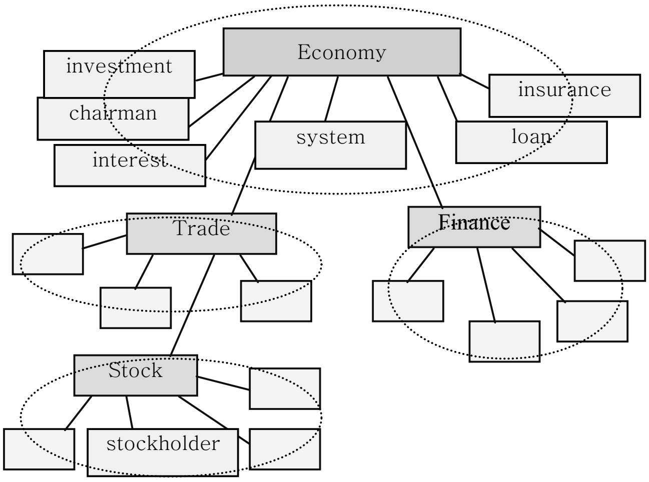


Figure 1: A portion of the domain ontology

3.3 Document Classification Using Ontology

The process of Web document classification basically involves two procedures: Finding key vocabulary in the documents and mapping onto a node in the concept hierarchy (ontology) using the extracted words.

As part of the key vocabulary extraction process from documents, the removal of inappropriate words and the stemming of words, both as the pre-classification procedures as well as the application of information retrieval measurement, *tf×idf ((*term frequency times inverted document frequency), take place. Stopping is a process of removing the most frequent word that exists in a web page document such as 'to', 'and', 'it', etc. Removing these words will save spaces for storing document contents and reduce time taken during the search process. Stemming is a process of extracting each word from a web page document by reducing it to a possible root word. For example, the words, 'compares', 'compared', and 'comparing' have similar meaning with the word 'compare'. After the stemming and stopping process of the erms in each document, we will represent them as the document-term frequency matrix ($Doc_j \times TF_{jk}$) as shown in Table 1. Doc_j is referring to each web page document that exists in the news database where $j=1,....,n$. Term frequency TF_{jk} is the number of how many times the distinct word w_k occurs in document Doc_j where $k=1,...,m$. The calculation of the terms weight x_{jk} of each word w_k is done by using a method that has been used by Salton [10] which is given by

$$x_{jk} = TF_{jk} \times idf_k \tag{1}$$

where the document frequency df_k is the total number of document in the database that contains the word w_k. The inverse document frequency $idf_k = \log(\frac{n}{df_k})$ where n is the total number of documents in the database.

Table 1. The document-term frequency data matrix after the stemming and stopping processes

Doc$_j$	**TF$_1$**	**TF$_2$**	**…**	**TF$_m$**
Doc$_1$	2	4	…	5
Doc$_2$	2	3	…	2
Doc$_3$	2	3	…	2
…	…	...	...	...
Doc$_n$	1	3	…	7

tf×*idf* is a mathematical algorithm which is used to efficiently find key vocabulary that best represents the texts by applying the term frequency and the inverted document frequency together.

tf(i,j) is the term frequency of Term *j* that appears in Document *di*∈*D**, where *i=1,2,3…,N*. *df(j)* is the document frequency of Term *j* and represents how often Term *j* appears in other documents. Then, *tf*×*idf* for Term *j* is defined as given below [11].

$$tf \times idf(i,j) = tf(i,j) \times \log\left(\frac{N}{df(j)}\right) \tag{2}$$

For vocabulary with a low or rare appearance frequency, the value of *tf*×*idf* is low, compared to that with a high appearance frequency, thus resulting words successfully classifying the documents. In the term selection process, a list of all terms contained in one document from the text collection is made. Then, the document selection process chooses Term *j* that maximizes *W(j)*, which is expressed as a vector for Document d_j as follows. Document d_j includes *tf*×*idf(i,j)*, which is *tf*×*idf* for the most appropriate term.

$$W(j) = \sum_{i=1}^{N} tf \times idf(i,j) \tag{3}$$

The similar calculation for classification is done using the following formula [12]: The text is mapped onto a node with the highest similarity value, and one text is ultimately classified into one class.

$$Sim(Node,d)=\frac{\sum_{i=0}^{N} freq_{i,d}/\max_{i,d}}{N}\times\frac{V_d}{V} \tag{4}$$

N is the feature frequency of a node. $freq_{i,d}$ represents the frequency of feature i that is matched in text d. $max_{i,d}$ is the frequency of the feature that is matched the most in Text d. V is the number of constraints, while V_d represents the number of constraints that are satisfied by Text d. The document classification takes place only when the use of the relations is "is-A", "has-A", "part-of", or "has-part". When another node is related, it is also included in the classification process to calculate the similarity. Using this approach, a more accurate classification can be performed.

The overall classification process for Web documents is as shown below (Figure 2).

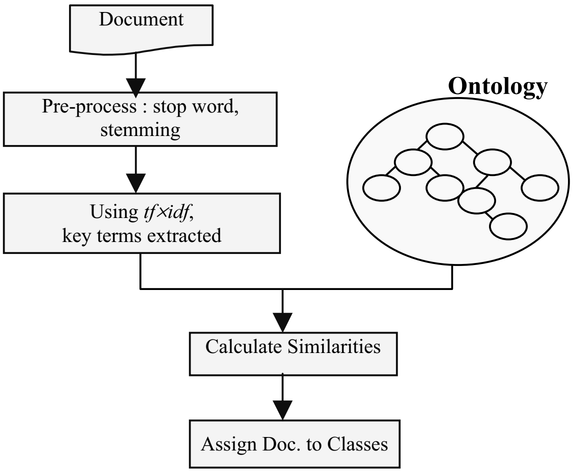

Document
Pre-process : stop word, stemming
Using *tf×idf*, key terms extracted
Ontology
Calculate Similarities
Assign Doc. to Classes

Figure 2: Web Document Classification Process Using Ontology

Ontology-based text categorization is the classification of documents by using ontologies as category definition.

In our approach, the process of text categorization is twofold: (1) Text categorization by calculating similarity between a feature vector and category vector, (2) Modifying weights between terms in an ontology by calculating similarity between category vectors (see Figure 3).

A feature vector is a vector which represents feature of a document, while a category vector is a vector which represents the characteristic of a category. The feature vector is calculated from the term frequency and the inverse document frequency. The category vector is calculated from the feature vector of the document assigned to the category.

We use vector space model commonly used in the information retrieval studies to weight terms and calculate feature vectors [10].

The document classification procedure starts from the ontology's root node as shown below Figure 3 and then goes to the sub nodes. The first step of the classification procedure is to calculate the feature vectors from the extracted vocabulary. The second step involves the distribution of the extracted vocabulary per the category vectors. The category vectors are again calculated using the distributed vocabulary in the third step. Then, the similarities between the feature vectors and category vectors are calculated and the documents are mapped onto the most appropriate node. The each initial category vector is calculated from the feature vector of the page which is assigned to the category by matching keywords. Using this technique, a small number of node knowledge can express the ontology of extracted vocabulary, making the document classification more accurate based on the hierarchical classification.

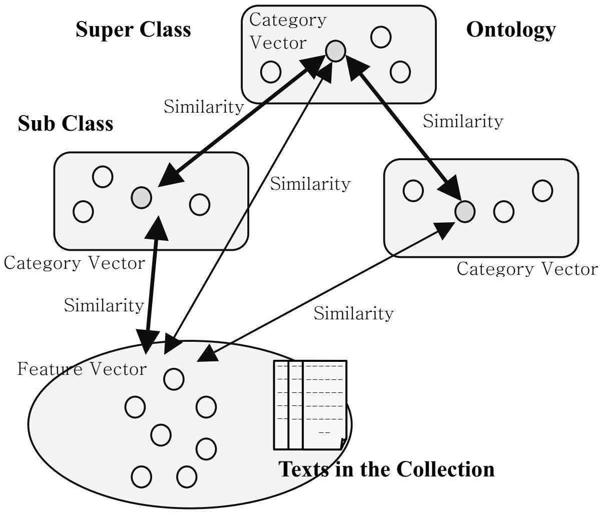


Figure 3. Document Classification Using Ontology

4. Experimental Procedures

We have used a web pages dataset from Yahoo Economy news as shown in Table 2. The types of news in the database are Cooperatives, Employment, Finance, Marketing, Organizations and Trade. The total of documents are 1,650.

Table 2. The number of document that are stored in the news database

Class no.	Class name	Number of documents
1	Cooperatives	22
2	Employment	685
3	Finance	336
4	Marketing	171
5	Organizations	133
6	Trade	303
	Total	1,650

(These are the classes that exist in Economy category.)

Automatic classification of web page is evaluated using the standard information retrieval measures that are precision, recall, and F1. They are defined as follows [13,14]

$$precision = \frac{a}{a+b} \tag{5}$$

$$recall = \frac{a}{a+c} \tag{6}$$

$$F1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \tag{7}$$

where the values of *a*, *b*, and *c* are defined in Table 3. The relationship between the system classification and the expert judgment is expressed using four values as shown in Table 4. The *F1* measure is a kind of average of precision and recall.

Table 3. The definitions of the parameters a, b, and c which are used in Table 4

	Meaning
a	The system and the expert agree with the assigned category
b	The system disagrees with the assigned category but the expert did
c	The expert disagrees with the assigned category but the system did
d	The system and the expert disagree with the assigned category

Table 4. The decision matrix for calculating the classification accuracies

	Expert	**System**	
	Yes	No	
Yes	a	b	a+b
No	c	d	c+d
	a+c	b+d	

Table 5. The classification results

Class no.	Precision (%)	Recall (%)	F1 (%)
1	77.21	93.84	84.72
2	92.48	94.16	93.31
3	93.93	95.38	94.65
4	91.17	95.38	93.23
5	91.30	96.92	94.03
6	91.97	96.92	94.38
Average	89.68	95.43	92.39

In Our the classification results, the precision, recall, and *F1* measures are 89.68%, 95.43%, 92.39%, respectively, as shown in Table 5.

A better document selection approach needs to be used for selecting the candidate documents from each class in order to increase the *F1* classification results.

5. Conclusion and Future Research

This paper introduced the use of ontology to conceptually express the meaning of relationships contained in Web documents and suggested an automated document classification method using the ontology. In particular, this paper focused on document classification based on the similarities of documents already categorized by ontology using terminology information extracted from the documents. Our work is distinguished from other studies in the following areas. (1) Rather than using a dictionary or knowledge index, ontology is used for document classification. (2) Our ontology is based on syntax information contained in the Web texts. (3) Mapping between the established ontology and the term information extracted from Web documents is performed. The document classification technique proposed by this paper does not involve any learning processes or experimental data and can be performed in real time.

Further research is required to develop more efficient and accurate ontological expressions and to document classification methods. We plan to conduct further studies on how to improve the efficiency of an information search using the document classification technique suggested in this paper and how to automatically determine the meaning of concepts and relations from Web documents.

References

[1] Chidanand Apt, Fred Damerau, and Sholom M. Weis, "Towards Language Independent Automated Learning of Text Categorization models," *Proc. of the 17th annual international ACM-SIGIR, 1994.*

[2] Jeuk Kim, Hanjoon Kim, Sanggoo Lee, "An Active Learning-based Method for Composing Training Document Set in Bayesian Text Classification Systems", *Korea Information Science Society Journals, Volume 29, 5th, 2002.12, Korea..*

[3] R.E.Shapire, Yoram Singhal, and Amit Singhal, "Boosting and Rocchio applied to text filtering,", *Proc. Of the 21th annual international ACM-SIGIR, 1998.*

[4] Mart A. Hearst, "Support Vector Machines," *IEEE Information Systems, 13(4):18~28, 1998.*

[5] Yiming Yang and Xin Liu, "A Re-examination of Text Categorization Methods", *Proc. Of the 22th annual International ACM-SIGIR, 1999.*

[6] T.R Gruber, "Towards Principles for the Design of Ontologies used for Knowledge Sharing,*" International Journal of Human-Computer Studies, 1995.*

[7] S.T.DUMAIS , and H.CHEN, "Hierarchical classification of Web content,", *Proc of the 23rd Annual International ACM SIGIR, July 24-28,2000, Arthens, Greece*.

[8] N.GOEVERT, M.LALMAS, and N.FUHR, 1999, "A probabilistic description-oriented approach for categorisiong Web documents,", *Proc. Of the 8th ACM International Conference on Information and Knowledge Management, November 2-4, 1999,pp 475-482, Kansas City, U.S.*

[9] Rudy Prabowo, Mike Jackson, Peter Burden, and Heinz-Dieter Knoell, 2002, "Ontology-Based Automatic Classification for the Web Pages : Design, Implementation and Evaluation,", *Proc. Of the 3rd International Conference on Web Information Systems Engineering, 2002.*

[10] Salton&McGill, Introduction to modern information retrival, New York, Mcgraw-Hill, USA, 1983.

[11] Andreas Hotho and Alexander Maedche and Steffen Staab, "Ontology-based Text Document Clustering", Http://www.aifb.uni-karlsruhe.de/WBS

[12] Hyunsup Jung, Jaeyoung Yang, Joongmin Choi, "An Ontology-based Recommendation Agent for Personalized Web Navigation", *Korea Information Science Society Journals Volume 30,1st. 2003. 2, Korea*.

[13] D.D.Lewis, Evaluating and optimizing autonomous text classification systems, in: E.A.Fox, P.Ingwersen, R.Fidel(Ed.), *SIGIR'95: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, 1995, pp. 246-254.

[14] Ali Selamat, Sigeru Omatu, Web page feature selection and classification using neural networks, *information sciences an international journal*, Vol 158(2004) pp, 69-88.

Performance Analysis of Naïve Bayes Classification, Support Vector Machines and Neural Networks for Spam Categorization

A. Cüneyd Tantuğ and Gülşen Eryiğit

{cuneyd, gulsen@cs.itu.edu.tr}
Dept. of Computer Engineering, Istanbul Technical Univ., 34469, Turkey

Abstract: Spam mail recognition is a new growing field which brings together the topic of natural language processing and machine learning as it is in essence a two class classification of natural language texts. An important feature of spam recognition is that it is a cost-sensitive classification: misclassification of a non-spam mail as spam is generally a more severe error than misclassifying a spam mail as non-spam. In order to be compared, the methods applied to this field should be all evaluated with the same corpus and within the same cost-sensitive framework. In this paper, the performances of Support Vector Machines (SVM), Neural Networks (NN) and Naïve Bayes (NB) techniques are compared using a publicly available corpus (LINGSPAM) for different cost scenarios. The training time complexities of the methods are also evaluated. The results show that NN has significantly better performance than the two other, having acceptable training times. NB gives better results than SVM when the cost is extremely high while in all other cases SVM outperforms NB.

Keywords: Spam Recognition, Support Vector Machines, Neural Networks, Naïve Bayes Classification

1 Introduction

Following the growth of the email usage, the unsolicited commercial electronic messages (spam) became a great annoying problem that costs billions of dollars. Spam mails waste users' time and bandwidth attacking the mailboxes and filling up file servers. The methods that can be used for building a spam filter can be chosen from 2-class classification techniques which are more reliable than today's keyword matching applications. There are a number of text classification techniques that can be adapted to spam filtering. But anti-spam filtering differs from other text classification tasks in two ways (Sakkis et al. 2003): first, the spam mails cover a wide spectrum of topics and second it is a cost-sensitive classification area. Obviously, marking a legitimate (non-spam) email as spam causes undesirable results; different costing schemes should exist in the classification methods. Each costing scheme represents an action taken when a spam mail is detected, i.e. delete it immediately, reserve it in the mailbox with a low priority, or send a verification mail to the sender. The more the action costs to the user, the more costing scheme penalize the classification errors. The spam mail filter, in another term the classifier, aim to classify a new incoming email as spam or legitimate (non-spam) according to the knowledge collected from the training stage of the classifier. Each email is treated as a pattern, which is represented by a feature vector whose elements are the occurrence flags of some selected words in this email. These words (attributes, in pattern recognition terminology) that characterize the emails are selected automatically according to their mutual information.

In this paper, three spam recognition methods are compared both under a non-cost-sensitive and a cost-sensitive framework. These methods are Support Vector Machines, experimented by Drucker et al. (1999) in a non-cost sensitive manner, Naïve Bayes experimented by Sakkis et al. (2003) in a cost-sensitive manner and Neural Networks. The results show that NN has significantly better performance than the two other having acceptable training times. NB gives better results than SVM when the cost is extremely high. In all other cases SVM outperforms NB.

Subsequent sections describe the related work, the details of the used corpus, the feature selection and the pattern representation methods to prepare the emails for the classification phase, the used classification methods and the comparison of the results obtained from each different method. In the final section, some conclusions and suggestions for future work are given.

2 Related Works

Spam filtering topic is a recent, important, and popular subject. The first technical work on automatic classification of spam and non-spam messages is on 1998 by Sahami et al.(1998). The followings are some of the classifiers developed so far: Naïve Bayes (Sahami et al. 1998; Androutsopoulos et al., 2000), Memory Based Learning (Sakkis et al. 2003), Boosting Trees (Carreras et al. 2001), Support Vector Machines (Drucker et al. 1999). In these classifiers, a supervised learning is implemented where the classification is fully automatic except the previously labeled training corpus. The previous studies (Sahami et al. 1998; Androutsopoulos et al., 2000, Sakkis et al., 2003; Carreras et al., 2001, Drucker et al., 1999) are not trained and tested on the same corpus. Thus, the results of these methods are out of usage if one tries to compare the methods in order to find the best one. Additionally, these results can hardly evaluated because not all of them are formulated within a cost-sensitive framework (Sakkis et al. 2003).

3 Corpus

An email corpus is needed in order to train the classifiers. Before initiating the training stage, all emails should be subject to some preprocessing tasks. These can be the removal of some most frequently used words using stop lists and the usage of a lemmatizer that finds the stem of the words. The only manual part of the classifiers is the preparation of the corpus where each mail is marked as legitimate or spam. One, who wants to filter emails in different languages or from different sources, should only prepare a relevant corpus. In this paper, the experiments are performed on a publicly available email corpus[1] which is a collection of spam and legitimate messages from a mailing list on linguistics "LINGSPAM" (Androutsopoulos et al. 2000). It should be known that the usage of mails from a mailing list will probably improve the filter's performance since an email list will be more topic-concentrated than individual mails. In "LINGSPAM", there exist four versions of the corpus which differ from each other by the usage of a lemmatizer and a stop-list. In this paper, the version with both lemmatizer and stoplist enabled is used during the experiments as it is emphasized in Androutsopoulos et al. (2000) that this version performs better for different cost criteria. The corpus consists of 2893 messages: 2412 legitimate and 481 spam. Ling-Spam is partitioned into 10 parts, with each part maintaining the same ratio of legitimate and spam messages as in the entire corpus. All of the experiments are done by utilizing 10-fold cross-validation mechanism, which means each test was repeated ten times, each time reserving a different part as the testing and using the remaining nine parts as the training corpus.

4 Feature Representation

The techniques are compared using multi-variate-model (Schneider 2003). That is, each mail is considered as a pattern which consists of features with binary values showing the presence or absence of a word in the current mail. Features correspond to words which are selected according to their mutual information calculated with the following formula (1) (Sahami et al. 1998; Androutsopoulos et al. 2000; Sakkis et al. 2003). "n" words with the biggest MI values are selected as the features and each mail is represented by the feature vector $\vec{x}$ =<x_1, x_2, …..,x_n>.

$$\mathbf{MI}(X,C) = \sum_{x\in\{0,1\},c\in\{spam,legitimate\}} P(X=x,C=c).\log_2\frac{P(X=x,C=c)}{P(X=x).P(C=c)} \tag{1}$$

[1] Ling-Spam available at http://www.aueb.gr/users/ion/

5 Classification Methods

In this section the classification techniques of NB, NN and SVM are given. Before going into details with these techniques, the cost-sensitive approach for the classification task is given. Mistakenly classifying a legitimate message as spam is generally more severe error than letting a spam message pass the filter that is classifying it as legitimate. Legit to Spam is λ time more costly than Spam to Legit: A mail is classified as spam if the following criterion (2) is met:

$$\frac{P(C = spam \mid \vec{X} = \vec{x})}{P(C = legitimate \mid \vec{X} = \vec{x})} > \lambda \tag{2}$$

In the case of email classification: P(C=spam| $\vec{X} = \vec{x}$) = 1-P(C=legitimate| $\vec{X} = \vec{x}$). As shown with the equations in (3), an instance $\vec{x}$ is classified as spam when the confidence level $W_S(\vec{x})$ is greater than t which is a function of λ.

$$\frac{P(C = spam \mid \vec{X} = \vec{x})}{1 - P(C = spam \mid \vec{X} = \vec{x})} > \lambda \tag{3}$$

$$W_s(\vec{x}) = P(C = spam \mid \vec{X} = \vec{x}) > t\,, \qquad t = \frac{\lambda}{\lambda + 1}$$

A cost scenario represents an action, for example if an email is classified as spam by the classifier, system's action can be just marking this email as "suspicious". In this case, cost value λ can be equal to 1 meaning that classifying an email as spam or legitimate has equal weights. If an email is deleted when it is classified as spam by the filter, the cost value must be very high, e.g. λ can be set to 999. In our work, three different scenarios are selected following the experiments in Sakkis et al. (2003), corresponding to λ=1, λ=9 and λ=999.

5.1 Neural Networks (NN)

An artificial neural network is a hierarchical organization, in which artificial neurons interacts with each other (Zurada 1992). Kohonen pointed out that artificial neural networks are hierarchical structures consisting of densely parallel connected adaptive components and having the ability to interact with real world like the biological systems do (Kohonen 1980).

A decisive property of artificial neural network is its learning ability. Learning of artificial neural network is realized through changing the connection weights of network, depending on input samples or in addition to them, depending on related outputs. Two types of learning strategies in literature can be classified as supervised learning and unsupervised learning (Efe 2000). The basic difference between them is the existence or nonexistence of desired outputs during learning procedure. The more and different samples are used in training, the more objects, artificial neural networks will recognize and the less error will be expected to occur.

The neural network used for spam filtering is implemented in MatLAB 6.5. The network type is chosen as 2-layer (10+2 neurons) feed-forward back propagation. The best performing learning function of NN is selected as gradient descent with momentum and adaptive learning rate back propagation. The class labels of training dataset (0 for spam, 1 for legitimate) are targets of NN and these labels are coded as [1 0] for non-spam class and [0 1] for spam class to improve NN classification performance.

5.2 Support Vector Machines

SVMs are powerful two-class pattern recognition methods which gains popularity since it was proposed by Vapnik (1995). The simple form of SVM is a linear machine which utilizes a preprocessing step for raising the dimensions of patterns. It is proven that, by a non-linear mapping to a sufficiently high dimension, the two classes can be separated by a hyperplane (Duda et al. 2001). Also, there are some modifications that transform a linear SVM into a non-linear classifier. SVM implementation needs quadratic programming to solve SVM QP problem. A number of methods exists in literature like Vapnik's "chunking" method (Vapnik 1982), Platt's Sequential Minimal Optimization–SMO (Platt 1998) and Osuna's theory (Osuna 1997). Further information about support vector machines can be found in (Vapnik 1995; Duda et al. 2001).

Although there are some tools which implements SVM (Torch, LibSVM), most of them lacks giving the confidence levels. In our work, we have used a library for SVM implementation called LibSVM (Chang et al. 2001). The latest version LibSVM 2.6 has the ability to give us the confidence levels of both classes, which allows us to compare SVM, NN and Naïve Bayes methods in a cost-sensitive framework. Linear kernel is chosen for solving the quadratic equations.

5.3 Naïve Bayes

From Bayes' theorem and the theorem of total probability, the probability that a document with vector $\vec{x}$ =<x1,x2,.….xn> belongs to category c is:

$$P\left(C=c \mid \vec{X}=\vec{x}\right)=\frac{P(C=c).P(\vec{X}=\vec{x} \mid C=c)}{\sum\limits_{k\,\in\,\{\text{spam, legitimate}\}} P(C=k).P(\vec{X}=\vec{x} \mid C=c)} \tag{4}$$

In practice, the probabilities $P(\vec{X}=\vec{x} \mid C=c)$ are impossible to estimate without making some assumptions, because the possible values of $\vec{x}$ are too many and there are also data sparseness problems. The Naïve Bayesian classifier assumes that x1,x2,.….xn are conditionally independent given the category c, which yields:

$$P\left(C = c \mid \vec{X} = \vec{x}\right) = \frac{P(C = c).\prod_{i=1}^{n} P(X_i = x_i \mid C = c)}{\sum_{k \in \{spam, legitimate\}} P(C = k).\prod_{i=1}^{n} P(X_i = x_i \mid C = c)} \quad (5)$$

6 Results

In this section, the results obtained from the implementation of SVM, NN and NB algorithms are given. The evaluation criteria is given in the equations (6 - 8) where $N_{x \to y}$ represents the number of x mails classified as y (x,y □ {**S**pam, **L**egitimate}) . Drucker et al. (1999) emphasizes that in two-class classification cases, recall and precision rates are useless, the false alarm rate and miss rate should be used instead. However, most of the previous works present their results by means of recall and precision rates. The cost function TCR (Total Cost Ratio) described in Sakkis et al. (2003) is suitable to be used in the comparison of the performances when the classification of a legitimate mail as spam is more costly than the classification of a spam mail as legitimate. The full derivation of TCR function can be found in Sakkis et al. (2003). "Greater TCR values mean better performance". It can be easily seen from the formula (8) that when TCR is less than 1, it is better to not use the filter at all. So, in our development, all of the five criteria are presented in order to obtain a relation with previous results.

$$FAR\ (False\ Alarm\ Rate) = \frac{N_{S \to L}}{N_S}, \quad MR\ (Miss\ Rate) = \frac{N_{L \to S}}{N_L} \quad (6)$$

$$RECALL\ \ (R) = \frac{N_{S \to S}}{N_{S \to S} + N_{S \to L}}, \quad PRESICION\,(P) = \frac{N_{S \to S}}{N_{S \to S} + N_{L \to S}} \quad (7)$$

$$TCR = \frac{N_S}{\lambda N_{L \to S} + N_{S \to L}} \quad (8)$$

Before the comparison of the methods, the attribute (feature) sizes that give the best results with the used corpus should be determined. Androutsopoulos et al. (2000) state that NB gives better results on LINGSPAM with the attribute dimensions dim=100 for λ=1, dim=100 for λ=9, dim=300 for λ=999. In our implementation, we observed that dim=100 for λ=999 gives also better results than dim=300. This can be seen in Fig. 1 where the performance evaluation with respect to attribute size is plotted. As there is no study in the literature that experiments LINGSPAM with SVM and NN, we experimented the SVM and NN for different attribute sizes varying from 100 to 700 stepped by 50 to find the optimum that gives the better TCR value. As can be seen from the Fig. 2, the maximum TCR values for each cost scenario are obtained from different attribute sizes. The attribute size is chosen as 600 for the following experiments since it gives the best TCR value in the average of the three different cost scenarios.

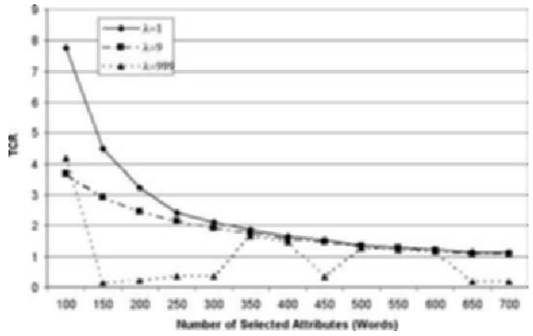


Fig. 1. Naïve Bayes TCR values / Dimensions

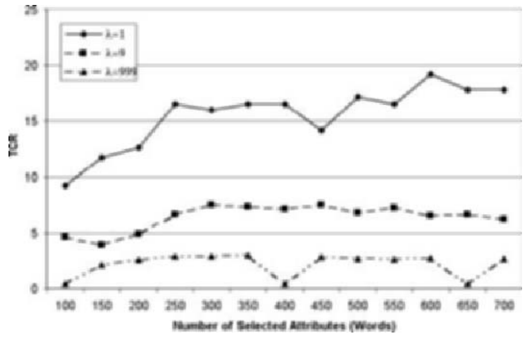


Fig. 2. SVM TCR values / Dimensions

The same tests are performed for the NN. The results are shown in Fig. 3. The best average TCR value can be achieved with a attribute size of 450. The next step of our study concerns about the training time complexity of these methods. NB has no training time because all of the information needed to calculate the probability distributions are gathered during the attribute selection step. Since we have used 10-fold cross-validation mechanism, we have measured the training time for each run. Thus, the methods are run 10-times for each attribute size. The averaged training times are given in Fig. 4.

Although, NN are said to be very time consuming in training phase (Drucker et al. 1999), our investigations show that, training time for NNs are in the same order with training time for SVMs. Also, the time necessary to train SVM and NN are linearly proportional to attribute size.

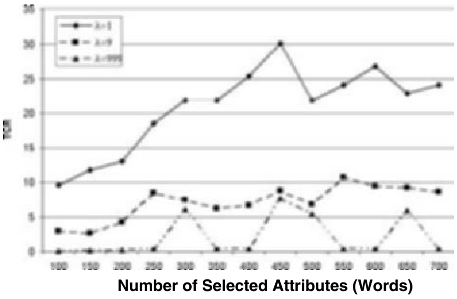


Fig. 3. NN TCR values / Dimensions

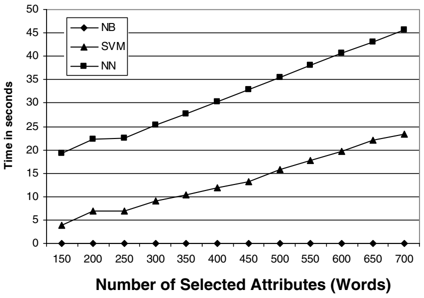


Fig. 4. Average training times for NB, NN and SVM per attribute size

6.1 Comparison of NB, NN and SVM

The methods are tested with the best attribute sizes observed and the results in terms of FAR and MR rates are given in Table-1. As MR (miss rate) increases, the number of misclassifications of legitimate emails increases while FAR (false alarm rate) increases, the number of misclassifications of spam emails (passing from the filter) increases. So both of FAR and MR should be as small as possible for an acceptable filter (should be 0 for a perfect filter). While considering the cost sensitive cases, the MR is more important and should be penalized more than FAR. Another evaluation criteria set is precision and recall rates. In Table-2, the precision and recall rates are presented per method and different cost values. It can be easily seen from Table-2 that NN performs best for all cases. The second place is shared between NB and SVM that SVM outperforms NB for λ=1 and λ=9 while NB outperforms SVM for λ=999. This can be more clearly seen from Table-3 which gives TCR values as a cost-sensitive measure of performances.

Table 1. False Alarm/Miss Rates

		NB (100)	SVM (600)	NN (450)
λ=1	FAR	0.1140	0.0350	0.0200
	MR	0.0029	0.0033	0.0025
λ=9	FAR	0.1600	0.1140	0.0400
	MR	0.0025	0.0008	0.0017
λ=999	FAR	0.2390	0.3600	0.1313
	MR	0.0000	0.0000	0.0000

Table 2. Precision/Recall Rates

		NB (100)	SVM (600)	NN (450)
λ=1	R	0.8856	0.9646	0.9792
	P	0.9838	0.9830	0.9874
λ=9	R	0.8399	0.8856	0.9605
	P	0.9853	0.9953	0.9914
λ=999	R	0.7609	0.6403	0.8690
	P	1.0000	1.0000	1.0000

Method	**λ=1 TCR**	**λ=9 TCR**	**λ=999 TCR**
NB(100)	7.77	3.68	4.19
SVM(600)	19.27	6.60	2.78
NN(450)	30.10	8.76	7.65

Table 3. Total Cost Ratio

7 Conclusions & Future Work

This paper aims to compare the performances of Naïve Bayes, Support Vector Machines and Neural Networks techniques in the field of spam mail recognition in a cost-sensitive framework. In order to benchmark the methods and use results of previous related works, a publicly available e-mail list (LINGSPAM) corpus is used. We have implemented NN, NB and SVM methods in a cost-sensitive manner and use a library for SVM. The evaluation is performed with three different cost scenarios. Results show that NN has significantly better performance than the two other, having acceptable training times. NB gives better results than SVM when the cost is extremely high while in all other cases SVM outperforms NB.

As a consequence, the contribution of our work to spam filtering tasks is the comparison of three methods on the same data set. As an additional contribution, NN and the cost-sensitive version of SVM are applied in the spam filtering subject. The neural networks were not preferred because of its time consuming training stage in spam recognition field. Our neural network implementation brings out surprising and promising results with lower error rates and acceptable training times.

Although spam mail filtering is performed by some pattern recognition techniques, not all of them are tested. Other methods should be implemented for spam filtering and compared with the others. Surprisingly some simple methods can achieve surprisingly spam filtering like Naïve Bayes. The linear kernel is used in our tests because a previous work (Drucker et al. 1999) uses this kernel but other kernel types (sigmoid, polynomial, etc.) should be examined and tested in order to find the most effective SVM classification.

The different representations of emails are not intensively investigated. Although a multi-variate representation scheme is chosen for keeping the relations with previous works, a multi-nomial representation can have a great impact on the performance of these methods. Investigating effects of other representations is covered by our future objectives. Moreover, some other data which gives clues about the email can be included in representation as domain attributes. In addition to selected words constituting the vector representation of emails, the existence of some word patterns like “FREE MONEY” can be new attributes. We plan to compare all possible methods and representation schemes in a cost-sensitive framework. We also plan to evaluate the same tests for different email corpora especially in different languages.

References

Androutsopoulos, I., Koutsias, J., Chandrinos, K.V., and Spyropoulos, C.D. (2000), "An Evaluation of Naive Bayesian Anti-Spam Filtering," *Proceedings of the workshop on Machine Learning in the New Information Age*, pp. 9-17.

Carreras, X., and Marquez, L. (2001), "Boosting Trees for Anti-Spam Email Filtering," *Proceedings of the 4th International Conference on Recent Advances in NLP*, pp. 58-64.

Chang, C.C, and Lin, C. (2001), "LIBSVM: a Library for Support Vector Machines," http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Duda, R.O., Hart, P.E., and Strok, D.G., (2001), "Linear Discriminant Functions," Chapter 5 in Pattern Classification. John Wiley, 10–43.

Drucker, H., Wu, D., and Vapnik, V.N., (1999), "Support Vector Machines for Spam Categorization," IEEE Transactions On Neural Networks, vol. 10(5), pp. 1048-1054.

Efe, O.M., and Kaynak, O. (2000), Artificial Neural Networks and their Applications, Bogaziçi University Press, Istanbul.

Kohonen, T. (1980), Content Addressable Memories, Springer-Verlag, New York

Osuna, E.E., Freund, R., and Girosi, F. (1997), "Improved training algorithm for support vector machines," Proceedings of the IEEE Workshops on Neural Network for Signal Processing, pp. 24-26.

Platt, J.C. (1998), "Sequential minimal optimization: A fast algorithm for training support vector machines," Advances in Kernel Method: Support Vector Learning, Scholkopf, Burges, and Smola, Eds. Cambridge, MA: MIT Press, pp. 185–208.

Sahami, M., Dumais, S., Heckerman, D., and Horvitz, E. (1998), "A Bayesian Approach to Filtering Junk E-Mail. Learning for Text Categorization," AAAI Technical Report, WS-98-05, pp. 55–62.

Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C.D., and Stamatopoulos, P. (2003), "A Memory-Based Approach to Anti-Spam Filtering for Mailing Lists," Information Retrieval, vol. 6(1), pp. 49-73.

Schneider, K. (2003), "A Comparison of Event Models for Naive Bayes Anti-Spam E-Mail Filtering," Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics, pp. 207-314.

Vapnik, V. (1995), The Nature of Statistical Learning Theory, Springer-Verlag.

Vapnik, V. (1982), Estimation of Dependences Based on Empirical Data, Springer-Verlag.

Zurada, J. M. (1992), Introduction To Artificial Neural Networks, West Publishing Company.

Sentence Extraction Using Asymmetric Word Similarity and Topic Similarity

M. Azmi-Murad and T.P. Martin

Department of Engineering Mathematics
University of Bristol
Bristol, BS8 1TR
UK
Email: {masrah.azmi-murad, trevor.martin}@bris.ac.uk

Abstract. We propose a text summarization system known as *MySum* in finding the significance of sentences in order to produce a summary based on asymmetric word similarity and topic similarity. We use mass assignment theory to compute similarity between words based on the basis of their contexts. The algorithm is incremental so that words or documents can be added or subtracted without massive re-computation. Words are considered similar if they appear in similar contexts, however, these words do not have to be synonyms. We also compute the similarity of a sentence to the topic using frequency of overlapping words. We compare the summaries produced with the ones by humans and other system known as TF.ISF *(term frequency-inverse sentence frequency*). Our method generates summaries that are up to 60% similar to the manually created summaries taken from DUC 2002 test collection.

Keywords: sentence extraction, asymmetric word similarity, topic similarity, mass assignment, fuzzy

1 Introduction

Information retrieval plays a significant role in web searching, which allows people access to various sources of information from anywhere and at anytime. A successful information retrieval system enables the user to determine quickly and accurately whether the contents of the documents are satisfactory. Nevertheless, the availability of large number of document makes it impossible for a user to view each one of them. What the users really need is an overview or basic ideas of the document in helping them to quickly evaluate the relevance of the document or navigate through a corpus.

Text summarization (Mani et al. 1999) is the process of compressing source text, for example, documents or books, by giving the final result as an overview or outline of content, at the same time maintaining the content originality and readability. A good summary will give clear information on the overall content of the document. Some examples on summaries are headlines of TV news, previews of movies, bulletins on weather forecasts and minutes of a meeting. The first automated sentence extraction system (Luhn 1958) uses term frequencies to weight sentences, which are then extracted to form an abstract. Since then, many approaches have been explored in automatically extracting sentences from a document.

Most of the existing summarization techniques use some variant of TF.IDF model (Salton and Buckley 1988). Among them (Larocca Neto et al. 2000), (Seki 2002), and (Lo et al. 2002). One summarization algorithm known as TF.ISF (term frequency-inverse *sentence* frequency) (Larocca Neto et al. 2000) is an adaptation of the conventional TF.IDF. The computation of TF.ISF is similar to the computation of TF.IDF with only one difference, in which the notion of *document* is replaced by *sentence*. Sentences with high values of TF.ISF are selected to produce a summary of the source text. Hence, the TF.ISF measure of a word *w* in a sentence *s*, is computed by the formula:

$$TF.ISF(w, s) = TF(w, s) * log(|S| / SF(w)) \qquad (1)$$

where *TF(w, s)* is the number of times the word *w* occurs in sentence *s*, *SF(w)* is the number of sentences in which the word *w* occurs and *S* is the total number of sentences in the document. For each sentence *s*, the average *TF.ISF* of the sentence is computed by calculating the average of the *TF.ISF(w, s)* weight over all of the words *w* in the sentence, as shown in the following formula:

$$Avg - TF.ISF = \sum_{i=1}^{W(s)} TF.ISF(i,s) \Big/ W(s) \qquad (2)$$

where *W(s)* is the number of words in the sentence *s*. Sentences with the largest values of *Avg-TF.ISF* are selected as the most relevant sentences and produce them as a summary. In this experiment, we will use TF.ISF as a comparison against our method.

2 Mass Assignments Theory and Fuzzy Sets

A fuzzy set (Zadeh 1965) is an extension to a classical set theory, which has a problem of defining the border of the set and non-set. For example, consider a height of a person with labels such as *tall, medium, short*. These labels are fuzzy because not everyone will agree with the same subset of the value domain as satisfying a given label. If they did, we could write precise definitions of *tall, medium* and *short* in this context.

A *mass assignment (MA)* theory (Baldwin et al. 1996) has been developed to provide a formal framework for manipulating both probabilistic and fuzzy uncertainties. For example, the voting model, suppose we have a set of people labeled 1 to 10 who are asked to accept or reject a dice value of *x* as *small*. Suppose everyone accepts 1 as *small*, 80% accept 2 as *small* and 20% accept 3 as *small*. Therefore, the fuzzy set for *small* is defined as:

$$small = 1 / 1 + 2 / 0.8 + 3 / 0.2 \quad (3)$$

where the membership value for a given element is the proportion of people who accept this element as satisfying the fuzzy set. The probability mass on the sets is calculated by subtracting one membership from the next, giving MA_{small} as:

$$MA_{small} = \{1\} : 0.2, \{1, 2\} : 0.6, \{1, 2, 3\} : 0.2 \quad (4)$$

The mass assignments above correspond to families of distribution. In order to get a single distribution, the masses are distributed evenly between elements in a set. This distribution is known as *least prejudiced distribution (LPD)* (Baldwin et al. 1996) since it is unbiased towards any of the elements. Thus, in the example above, the mass of 0.6 is distributed equally among 1 and 2 and the mass 0.2 is distributed equally among 1, 2 and 3. Therefore, the *least prejudiced distribution* for *small* is:

$$LPD_{small} = 1 : 0.2 + 0.3 + 0.0667 = 0.5667, \\ 2 : 0.3 + 0.0667 = 0.3667, 3 : 0.0667 \quad (5)$$

2.1 Semantic Unification

Semantic unification (Baldwin et al. 1996) is the process of calculating conditional probabilities of fuzzy sets. A mass assignment with the *least prejudiced distribution* can be used to determine the point semantic unification. For example, suppose the fuzzy set for *medium* in the voting model is:

$$medium = 2 / 0.2 + 3 / 1 + 4 / 1 + 5 / 0.2 \quad (6)$$

and the mass assignment would be:

$$MA_{medium} = \{3, 4\} : 0.8, \{2, 3, 4, 5\} : 0.2 \quad (7)$$

Thus, the *least prejudiced distribution* is:

$$LPD_{medium} = 2 : 0.05, 3 : 0.45, 4 : 0.45, 5 : 0.05 \quad (8)$$

Suppose we want to determine the *Pr(about_3|medium),* and the fuzzy set is:

$$about\_3 = 2 / 0.4 + 3 / 1 + 4 / 0.4 \quad (9)$$

with mass assignment as:

$$MA_{about\_3} = \{3\} : 0.6, \{2, 3, 4\} : 0.4 \quad (10)$$

The point semantic unification can be calculated using the following tableau.

Table 1. Tabular Form of the Pr(about_3|medium)

	0.8 : {3, 4}	**0.2 : {2,3,4,5}**
0.6 : {3}	1/2 x 0.8 x 0.6	1/4 x 0.2 x 0.6
0.4 : {2, 3,4}	0.8 x 0.4	3/4 x 0.2 x 0.4

The entries in the cells are the supports from the individual terms of the mass assignments. The masses are multiplied and a proportion of the product taken where the proportion represents the number of elements of the associated set of the mass assignment of the given information in the associated set of the mass assignment on the left. Thus, 3 of the elements of {2, 3, 4, 5} are in {2, 3, 4}, so we take the proportion as 3/4. The final value of the *Pr(about_3|medium)* is 0.65. The computation of the probability above can be shown using the following formula. Consider two fuzzy sets *f1* and *f2* defined on a discrete universe *X*. Let:

$\mu_{f1}(x)$ be the membership of element *x* in the fuzzy set *f1*.
$MA_{f1}(S)$ be the mass associated with set *S*.
$LPD_{f1}(x)$ be the probability associated with element *x* in the *LPD*.

Then:

$$Pr(f1|f2) = \sum_{\substack{S1 \subseteq X, \\ S2 \subseteq X, \\ S1 \cap S2 \neq \phi}} \frac{MA_{f1}(S1) \times MA_{f2}(S2)}{|S2|} \quad \textbf{(11)}$$

which can be re-written as:

$$Pr(f1|f2) = \sum_{x \in X} \mu_{f1}(x) \times LPD_{f2}(x) \quad \textbf{(12)}$$

3 Computation of Similarity

In this section, we present an algorithm for finding word relations for the purpose of extracting significance sentences. We also present an algorithm for computing sentence-topic similarity. We experiment with data taken from DUC 2002 in producing summaries. We apply AWS in the document to compute sentence-sentence similarity. Our method concentrates on how sentences use a word, *w*, and not in their meaning. Thus, words are considered similar if they appear in similar contexts.

3.1 Word similarity

As mentioned in (Harris 1985), words that tend to occur in documents in similar contexts tend to have similar meanings. Consider the following example, taken from (Pantel and Lin 2002):

A bottle of *tezgüno* is on the table.
Everyone likes *tezgüno*.
Tezgüno makes you drunk.
We make *tezgüno* out of corn.

From the sentences above, we could infer that *tezgüno* may be a kind of an alcoholic beverage. This is because other alcoholic beverage tends to occur in the same contexts as *tezgüno.* This assumption is based on a principle known as the Distributional Hypothesis (Harris 1985).

We use this idea to produce a set of related words, which can be used as the basis for taxonomy, or to cluster documents. In this experiment, we use Fril (Baldwin et al. 1995) to compute asymmetric similarity such that the similarity between <*term1*> and <*term2*> is not necessarily the same as between <*term2*> and <*term1*>. Thus:

$$ws(<term1>,< term2>) \neq ws(<term2>, <term1>)$$

This is because to compute similarity between two fuzzy sets, i.e. *ws(<term1>,< term2>),* we will multiply the memberships of fuzzy sets of <*term1*> with the corresponding frequencies in frequency distributions of <*term2*>. In order to calculate *ws(<term2>, <term1>),* we will multiply the memberships of fuzzy sets of <*term2*> with the corresponding frequencies in frequency distributions of <*term1*>. In most cases, the number of elements for two fuzzy sets is different; therefore, the similarity measures will be different. Our method is based on finding the frequencies of *n-tuples* of context words in a set of documents. Consider the sentences below:

The quick brown fox jumps over the lazy dog.
The quick brown cat jumps onto the active dog.
The slow brown fox jumps onto the quick brown cat.
The quick brown cat leaps over the quick brown fox.

For each word *x*, we incrementally build up a set *context-of-x* containing pairs of words that surround *x,* with a corresponding frequency, e.g. for *brown* we have *(quick, cat)* occurs three times, *(quick, fox)* occurs two times and *(slow, fox)* occurs once. We use mass assignment theory to convert these frequencies to fuzzy set. Thus, we will have the membership as:

$$\{ (quick, cat):1, (quick, fox):0.833, (slow, fox):0.5 \}$$

For any two words <*term1*> and <*term2*>, Pr(<*term1*>|<*term2*>) measures the degree to which <*term1*> could replace <*term2*>, and can be calculated by semantic unification of the two fuzzy sets characterising their contexts. If two terms are fuzzy sets, a point value is calculated for <*term1*> given <*term2*>. The order of term is important, since the support of <*term1*> given <*term2*> will differ than support of <*term2*> given <*term1*>. For example, suppose there are sentences in the document that give the fuzzy context set of *grey* as:

$$\{(quick, cat):1, (slow, fox):0.75\}$$

We calculate:

$$Pr(brown|grey) = 0.8125$$
$$Pr(grey|brown) = 0.625$$

This gives an asymmetric word similarity matrix $M^{(w)}$, whose rows and columns are labelled by all the words encountered in the document collection. Each cell $M^{(w)}$ *(i, j)* holds a value between 0 and 1, indicating to which extent a word *i* is contextually similar to word *j*. For any word we can extract a fuzzy set of similar words from a row of the matrix. Many of the elements are zero. We also note that there are important efficiency considerations in making this a totally incremental process, i.e. words (and documents) can be added or subtracted without having to recalculate the whole matrix of values.

As would be expected, this process gives both sense and nonsense. Related words appear in the same context (as with *grey* and *brown* in the illustration above), however, unrelated words may also appear, e.g. the phrase *slow fat fox* would lead to a non-zero similarity between *fat* and *brown*.

Table 2 shows some selection of asymmetric similarity between words taken from a document set, D061 in the DUC 2002 test collection. Most word pairs are plausible although some, for example, *hit* and *decrease* may be dissimilar. As mentioned earlier, the asymmetric word similarity is purely due to similar contexts, and may not lead to similar meaning. The probability of two words is calculated based on two fuzzy sets characterising their contexts.

Table 2. The fuzzy approach as a measure of contextual word similarity

Word similarity matrix			
word i	*word j*	$ws(w_i,w_j)$	$ws(w_j,w_i)$
hit	decrease	1	0.2
rain	flood	0.072	0.062
approach	mile	0.058	0.333
watch	warn	0.111	0.155
evacuate	homeless	1	0.125
warn	precaution	1	0.222
strongest	intense	0.667	0.5
nation	Cuba	0.084	0.053
province	bay	0.5	0.2
today	Monday	0.2	0.333

We use the word similarity matrix in computing sentence-sentence similarity. Hence, the pairwise sentence similarity measure is calculated using the following formula:

$$Sim(S_i, S_j) = \sum_i \sum_j ws(w_{ai}, w_{bj}).w_{ai}.w_{bj} \qquad \textbf{(13)}$$

where $ws(w_{ai}, w_{bj})$ is the word similarity of two fuzzy sets, w_{ai} and w_{bj} are the normalized weight of word *a* and word *b* in sentence *i* and *j* respectively. This method also applies for asymmetric sentence similarity using asymmetric word similarity, which would produce a different similarity measure.

3.2 Topic Similarity

The purpose of computing topic similarity is to increase the importance measure of a sentence S_i to the topic. We compute weight for topic similarity using the frequency of overlapping words in the sentence as well as the topic. Identical words will have a value of 1, while 0 for non-identical words. Hence, the formula is given as below:

$$Sim(S_i, t) = \sum_i \sum_t sim(w_{ai}, w_{bt}).w_{ai}.w_{bt} \tag{14}$$

where $sim(w_{ai}, w_{bt})$ is the similarity of two identical words and w_{ai} and w_{bt} are the normalized weight of word *a* and *b* in sentence *i* and topic *t.*

4 Sentence Extraction

We use the two score functions, i.e., word similarity and topic similarity to compute weight for each sentence. We measure the importance of a sentence S_i as an average similarity *Avgsim(i),* of which the weight of a sentence is defined by summing similarity measure of sentence S_i with other sentences in the document, divided by *N* total number of sentences. Thus, the *Avgsim(i)* is computed by the formula:

$$Avgsim(i) = \sum_{\substack{j=1 \\ j \neq i}} Sim(S_i, S_j) \Big/ N \tag{15}$$

where $Sim(S_i, S_j)$ is the pairwise asymmetric sentence similarity and *N* is the total number of sentences available in the document. Next, we add the weight of average sentence similarity and topic similarity to produce the final score of a sentence as shown in the formula:

$$MySum = Avgsim(i) + Sim(S_i, t) \tag{16}$$

After the score of each sentence is computed, the sentences are ranked in descending order. Sentences with high values will be selected to produce a summary. Then the sentences are arranged according to their chronological order in the original article to form a summary.

5 Experimental Results

In order to evaluate the *MySum* algorithm, we compare the summaries produced by our system against the manually created summaries produced in the DUC 2002. In addition, we also compare the performance of other system, TF.ISF against the manually created summaries. Our final comparison is between *MySum* and TF.ISF against the manually created summaries. Each document produces two versions of manually created summaries written by two different human readers. In this experiment, we produce a hypothetical testing by making comparison of summaries produced by two different human summarizers. This is to show that in reality it is very unlikely for two different systems or human to produce an identical summary from a document.

5.1 DUC Collection

In DUC 2002 there are about 59 document sets that consist of 567 related newswire documents from the Associated Press, Wall Street Journal, Financial Times, LA Times and San Jose Mercury News, spanning the period from 1988 to 1992. The average length of a document is 30 sentences. We choose randomly eight document sets from the corpus that consist of 68 newswire documents in total and use them in our experiment. In DUC 2002, Task 1 is a single-document summarization in which the goal is to extract 100 word summaries from each document in the corpus. We use the summaries produced in Task 1 in our comparison stage. The comparison is made using individual matching, i. e., each sentence in a summary produced by *MySum* or TF.ISF is compared against each sentence of the manually created summaries. Identical summaries will have a value of 1. The figure below shows the comparison of average similarity produced by *MySum* and TF.ISF against human summarizers, P1 and P2. In the figure, the comparison of two summaries produced by P1 and P2 is used as a hypothetical testing.

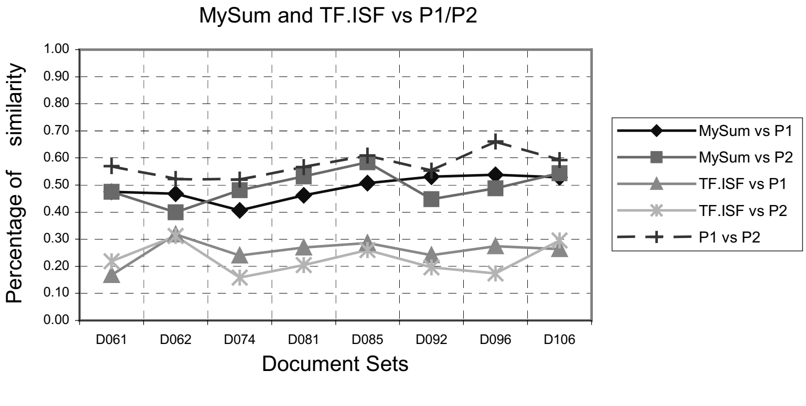


Fig. 1. Average similarity on summaries comparison

On average, *MySum* produces summaries that are up to 60% similar to the manually created summaries, while TF.ISF produces summaries that are up to 30% similar. In addition, the human summarizers, P1 and P2 produce summaries that is up to 70% similar. Overall, *MySum* produces a fairly good result and none of the documents generated by *MySum* produce a NIL similarity comparison against the manually created summaries. Our system shows that it could generate a summary from a document as close to what a human summarizer could produce. In Table 3, we show a few text summaries generated by *MySum* and human summarizer taken randomly from the document sets. Summaries generated by *MySum* are generally longer because of the original sentences extracted from the document.

Table 3. Example on summaries generated by *MySum* and human summarizer

No	*MySum*	*Manually created summary*
1	The San Francisco 49ers routed the Denver Broncos 55-10 Sunday in the most lopsided Super Bowl victory ever. The 49ers' win in the 24th Super Bowl made them the first repeat NFL champion in a decade and tied the Pittsburgh Steelers as a pinnacle of Super Bowl perfection with four wins in four tries. The Broncos, on the other hand, lost the last four Super Bowl games they have played. San Francisco quarterback Joe Montana made five touchdown passes, three to Jerry Rice breaking a Super Bowl record for touchdown passes on a day on which he also set a record with Valuable Player award and San Francisco's point total was the most ever. In four Super Bowls, he has thrown 11 touchdowns and no interceptions. For Denver quarterback John Elway, it was a day of futility, ending with his third Super Bowl defeat.	The San Francisco 49ers routed the Denver Broncos 55-10 Sunday in the most lopsided Super Bowl victory ever. This was the 49ers fourth win in four tries, tying them with the Pittsburgh Steelers for the NFL championship. San Francisco quarterback Joe Montana set several Super Bowl records; five touchdown passes; 13 straight pass completions; a third Super Bowl MVP award. San Francisco's point total was the most ever. In four Super Bowls, Montana has thrown 11 touchdowns and no interceptions. It was a day of futility for Broncos' quarterback John Elway as his team suffered its third Super Bowl defeat.
2	A one-day strike by coal miners in Kazakhstan and the Ukraine has grown into a nationwide walkout involving up to 100,000 miners demanding that President Gorbachev surrender power to the restive republics, a union leader said Friday. Most of the country's 1.2 million coal miners are still working, but output in some key regions has been halved, saddling Gorbachev with a substantial labour and fuel problem even as he struggles to revive the Soviet economy. They claim the government has not kept promises it made to raise miners' living standards as settlement of a nationwide coal strike in July 1989. The 1989 strike, which involved more than 500,000 miners, made only economic demands. This time, Shushpanov said, the miners also want three major political concessions.	A one-day strike by coal miners in Kazakhstan and the Ukraine has grown into a nationwide walkout involving up to 100,000 miners demanding that President Gorbachev surrender power to the restive republics. Most of the country's 1.2 million coal miners are still working, but output in some key regions has been halved. The strike is the new union's first major test of strength. It began as a warning in Kazakhstan and the Ukraine but gradually spread to all of the country's major regions. Previous strikes have made only economic demands. This time the miners want political concessions.
3	The White House is making sure nobody will accuse it of taking this crisis lightly. In the aftermath of the California earthquake, President Bush and his aides flew into a whirlwind of earthquake-related activity yesterday morning. So the White House announced that Mr. Bush got his first earthquake briefing of the day at 6:30 a.m. from chief of staff John Sununu. Mr. Bush himself essentially acknowledged that he and his aides were trying to head off criticism. On his FEMA visit, Mr. Bush said that he hoped there would be "less carping" about the emergency office's performance this time, adding that the agency "took a hit" for its reaction to Hurricane Hugo. The White House already is talking of Mr. Bush visiting the California earthquake site this weekend.	Yesterday, in response to the California earthquake, President Bush and his aides flew into a whirlwind of activity that seemed designed to dispel any thought that the White House was not engaged. After the Exxon Valdez oil spill and the devastation of Hurricane Hugo the administration was criticized for its slow response. Bush as much as admitted that the present flurry of activism was intended to counter such criticism, remarking during a visit to the Federal Emergency Management Agency that he hoped there would be "less carping" this time about the Agency's performance. The President may visit the earthquake site this weekend.

6 Conclusions

In this experiment, we proposed a new algorithm, *MySum* to extract sentences from a document. Experiments show that using the combination of both the word similarity and topic similarity is able to extract the most important sentences from a document that is fairly close to the manually created summaries. Although *MySum* did not produce a similar summary to the one created by human, on average, *MySum* is able to give a representable extractive summary. On the other hand, *MySum* outperforms TF.ISF when compared against the manually created summaries. One advantage of using *MySum* is that it can be used in multi-document summarization.

In addition, we are currently testing *MySum* with longer documents to ensure that it will produce a consistently satisfactory result.

References

Baldwin JF, Martin TP, Pilsworth BW (1995) *Fril-Fuzzy and Evidential Reasoning in Artificial Intelligence*. Research Studies Press, England

Baldwin JF, Martin TP, Pilsworth BW (1996) "A Mass Assignment Theory of the Probability of Fuzzy Events," *Fuzzy Sets and Systems*, (83), pp 353-367

DUC (2002) DUC-Document Understanding Conferences, http://duc.nist.gov

Harris Z (1985) Distributional Structure. In: Katz JJ (ed) *The Philosophy of Linguistics*. New York: Oxford University Press, pp 26-47

Larocca Neto J, Santos AD, Kaestner CAA, Freitas AA (2000b) Document clustering and text summarization. In *Proc. 4th Int. Conf. Practical Applications of Knowledge Discovery and Data Mining (PADD-2000)*, London: The Practical Application Company, pp 41–55

Lo SH, Meng H, Lam W (2002) "Automatic Bilingual Text Document Summarization, " *Proceedings of the Sixth World Multiconference on Systematic, Cybernetics and Informatics*, Orlando, Florida, USA

Luhn H (1958) The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(92):159–165

Mani I, Maybury MT (eds) (1999) *Advances in Automatic Text Summarization*, Cambridge, MA: The MIT Press

Pantel P, Lin D (2002) "Discovering Word Senses from Text," In *Conference on Knowledge Discovery and Data Mining*, Alberta, Canada

Salton G, Buckley C (1988) Term-weighting approaches in automatic text retrieval. *Information Processing and Management 24*, pp 513-523. Reprinted in: Sparck Jones K. and Willet P. (eds) (1997) *Readings in Information Retrieval*, Morgan Kaufmann, pp 323-328

Yohei S (2002) "Sentence Extraction by tf/idf and Position Weighting from Newspaper Articles TSC-8)," *NTCIR Workshop 3 Meeting TSC*, pp 55-59

Zadeh LA (1965) "Fuzzy Sets," *Information and Control*, vol. 8, pp 338-353

Part XIII

Algorithm Design

Designing Neural Networks Using Gene Expression Programming

Cândida Ferreira
Gepsoft, 73 Elmtree Drive
Bristol BS13 8NA, United Kingdom

Abstract. An artificial neural network with all its elements is a rather complex structure, not easily constructed and/or trained to perform a particular task. Consequently, several researchers used genetic algorithms to evolve partial aspects of neural networks, such as the weights, the thresholds, and the network architecture. Indeed, over the last decade many systems have been developed that perform total network induction. In this work it is shown how the chromosomes of Gene Expression Programming can be modified so that a complete neural network, including the architecture, the weights and thresholds, could be totally encoded in a linear chromosome. It is also shown how this chromosomal organization allows the training/adaptation of the network using the evolutionary mechanisms of selection and modification, thus providing an approach to the automatic design of neural networks. The workings and performance of this new algorithm are tested on the 6-multiplexer and on the classical exclusive-or problems.

Keywords: neural networks, gene expression programming, evolvable neural networks

Introduction

An artificial neural network is a computational device that consists of many simple connected units (neurons) that work in parallel. The connections between the units or nodes are weighted usually by real-valued weights. Weights are the primary means of learning in neural networks, and a learning algorithm is used to adjust the weights (e.g., Anderson 1995).

More specifically, a neural network has three different classes of units: input, hidden, and output units. An activation pattern is presented on its input units and spreads in a forward direction from the input units through one or more layers of hidden units to the output units. The activation coming into a unit from other units is multiplied by the weights on the links over which it spreads. All incoming activation is then added together and the unit becomes activated only if the incoming result is above the unit's threshold.

In summary, the basic elements of a neural network are the units, the connections between units, the weights, and the thresholds. And these are the elements that must be encoded in a linear chromosome so that populations of such structures can adapt in a particular selection environment in order to evolve solutions to different problems.

Over the last decade many systems have been developed that evolve both the topology and the parametric values of a neural network (Angeline et al. 1993, Braun and Weisbrod 1993, Dasgupta and McGregor 1992, Gruau et al. 1996, Koza and Rice 1991, Lee and Kim 1996, Mandischer 1993, Maniezzo 1994, Opitz and Shavlik 1997, Pujol and Poli 1998, Yao and Liu 1996, Zhang and Muhlenbein 1993). The present work introduces a new algorithm, GEP-NN, based on Gene Expression Programming (GEP) (Ferreira 2001) that performs total network induction using linear chromosomes of fixed length (the genotype) that map into complex neural networks of different sizes and shapes (the phenotype). The problems chosen to show the workings of this new algorithm include two problems of logic synthesis: the exclusive-or and the 6-multiplexer.

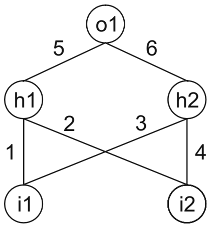
Genes with Multiple Domains for Designing NNs

The total induction of neural networks (NNs) using GEP, requires further modification of the structural organization developed to manipulate numerical constants (Ferreira 2001, 2003). The network architecture is encoded in the familiar structure of head and tail. The head contains special functions that activate the units and terminals that represent the input units. The tail contains obviously only terminals. Besides the head and the tail, these genes (neural net genes or NN-genes) contain two additional domains, Dw and Dt, encoding, respectively, the weights and the thresholds. Structurally, the Dw comes after the tail and has a length d_w equal to the head length h multiplied by maximum arity n, and Dt has a length d_t equal to h. Both domains are composed of symbols representing the weights or thresholds of the neural net.

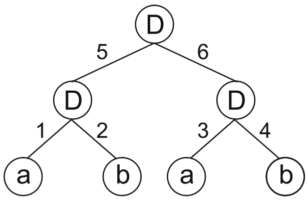
For each NN-gene, the weights and thresholds are created at the beginning of each run, but their circulation is guaranteed by the usual genetic operators of mutation, transposition, and recombination. Nonetheless, a special mutation operator was created that allows the permanent introduction of variation in the set of weights and thresholds.

It is worth emphasizing that the basic genetic operators like mutation or transposition are not affected by Dw and Dt as long as the boundaries of each region are maintained and the alphabets of each domain are not mixed up.

Consider the conventionally represented neural network with two input units (i_1 and i_2), two hidden units (h_1 and h_2), and one output unit (o_1) (for simplicity, the thresholds are all equal to 1 and are omitted):



It can also be represented as a tree:



where *a* and *b* represent, respectively, the inputs i_1 and i_2 to the network and “D” represents a function with connectivity two. This function multiplies the value of each argument by its respective weight and adds all the incoming activation in order to determine the forwarded output. This output (0 or 1) depends on the threshold which, for simplicity, was set to 1.

We could linearize the above NN-tree as follows:

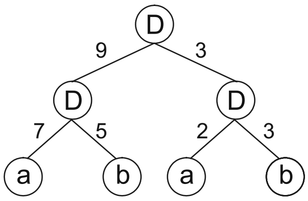
```
0123456789012
DDDabab654321
```

which consists of an NN-gene with the familiar head and tail domains, plus an additional domain Dw for encoding the weights. The values of each weight are kept in an array and are retrieved as necessary. For simplicity, the number represented by the numeral in Dw indicates the order in the array.

Let us now analyze a simple neural network encoding a well-known function, the exclusive-or. Consider, for instance, the chromosome below with $h = 3$ and containing a domain encoding the weights:

```
0123456789012
DDDabab393257
```

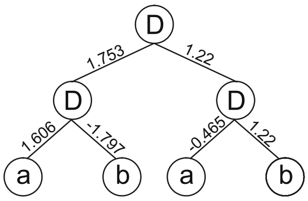
Its translation gives:



For the set of weights:

W = {-1.978, 0.514, -0.465, 1.22, -1.686, -1.797, 0.197, 1.606, 0, 1.753},

the neural network above gives:



(2.1)

which is a perfect solution to the exclusive-or problem.

Special Genetic Operators

The evolution of such complex entities composed of different domains and different alphabets requires a special set of genetic operators so that each domain remains intact. The operators of the basic gene expression algorithm (Ferreira 2001) are easily transposed to neural-net encoding chromosomes, and all of them can be used as long as the boundaries of each domain are maintained and alphabets are not mixed up. Mutation was extended to all the domains and continues to be the most important genetic operator. IS and RIS transposition were also implemented in GEP-nets and their action is obviously restricted to heads and tails. However, a special insertion operator was created that operates within Dw and Dt, ensuring the efficient circulation of weights and thresholds in the population. Another special operator, weights and thresholds' mutation, was also created in order to directly introduce variation in the set of available weights and thresholds.

The extension of recombination and gene transposition to GEP-nets is straightforward, as their actions never result in mixed domains or alphabets. However, for them to work efficiently (i.e., allow an efficient learning), we must be careful in determining which weights and/or thresholds go to which region after the splitting of the chromosomes, otherwise the system is incapable of evolving efficiently. In the case of gene recombination and gene transposition, keeping track of a gene's weights and thresholds is no difficult task, and these operators work very well in GEP-nets. But in one-point and two-point recombination where chromosomes can be split anywhere, it is impossible to keep track of the weights and thresholds. In fact, if applied straightforwardly, these operators would produce such evolutionary monsters that they would be of little use in multigenic chromosomes. Therefore, for multigenic systems, a special intragenic two-point recombination was created so that the recombination is restricted to a particular gene.

Domain-specific Transposition

Domain-specific transposition is restricted to the NN-specific domains, Dw and Dt. Its mechanism is, however, similar to IS transposition (Ferreira 2001). This operator randomly chooses the chromosome, the gene with its respective Dw plus Dt (if we use the same symbols to represent the weights and the thresholds, we can treat Dw and Dt as one big domain), the first position of the transposon, the transposon length, and the target site (also chosen within Dw plus Dt).

Consider the chromosome below with h = 4 (Dw and Dt are shown in different shades):

01234567890123456**7890123456789012**3456

DTQaababaabbaabba**0571745736284668**2867 (3.1)

where "T" represents a function of three arguments and "Q" represents a function of four arguments. Suppose that the sequence "46682" was chosen as a transposon and that the insertion site was bond 4 in Dw (between positions 20 and 21). Then the following chromosome is obtained:

01234567890123456**7890123456789012**3456

DTQaababaabbaabba**0571466827457362**8466 (3.2)

Note that the transposon might be any sequence in Dw or Dt, or even be part Dw and part Dt like in the example above. Note also that the insertion site might be anywhere in Dw or Dt as the symbols used to represent the weights and the thresholds are the same. Remember, however, that the values they represent are different for they are kept in different arrays. Suppose that the arrays below represent the weights and the thresholds of chromosome (3.1) above:

W = {-1.64, -1.834, -0.295, 1.205, -0.807, 0.856, 1.702, -1.026, -0.417, -1.061}
T = {-1.14, 1.177, -1.179, -0.74, 0.393, 1.135, -0.625, 1.643, -0.029, -1.639}

Although the new chromosome (3.2) obtained after transposition has the same topology and uses exactly the same arrays of weights and thresholds for its expression, a different neural network is encoded in this chromosome (Figure 1). Indeed, with domain-specific transposition the weights and thresholds are moved around and new combinations are tested.

Intragenic Two-point Recombination

Intragenic two-point recombination was created in order to allow the modification of a particular gene without interfering with the other sub-NNs encoded in other genes. The mechanism of this kind of crossover is exactly the same as in two-point recombination, with the difference that the crossover points are chosen within a particular gene (see Figure 2).

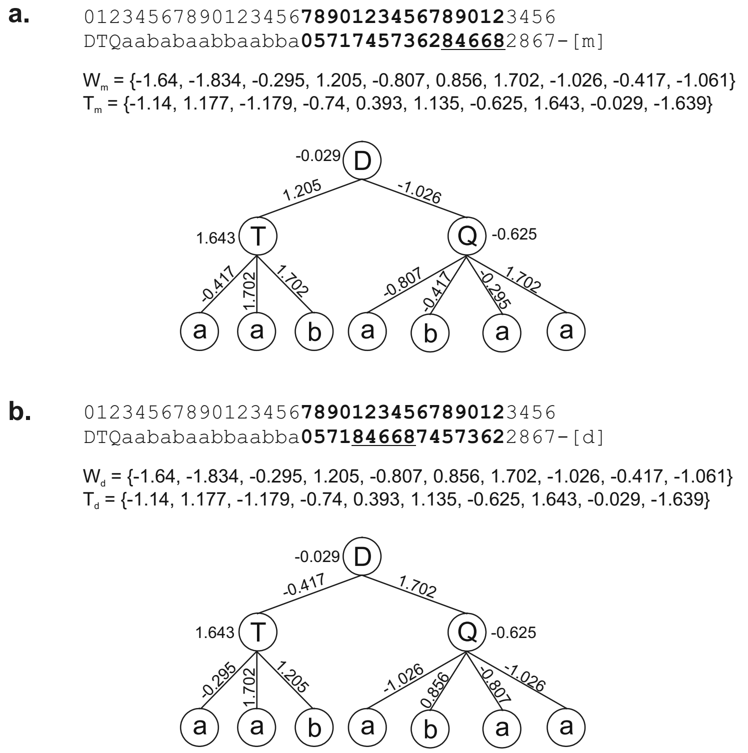
Consider the following parent chromosomes composed of two genes, each with a weights' domain ($W_{i,j}$ represents the weights of gene *j* in chromosome *i*):

$W_{0,1}$ = {-0.78, -0.521, -1.224, 1.891, 0.554, 1.237, -0.444, 0.472, 1.012, 0.679}
$W_{0,2}$ = {-1.553, 1.425, -1.606, -0.487, 1.255, -0.253, -1.91, 1.427, -0.103, -1.625}

```
0123456789012345601234567890123456
TTababaab14393255QDbabbabb96369304-[0]
Qaabbbabb97872192QDbabbaaa81327963-[1]
```

$W_{1,1}$ = {-0.148, 1.83, -0.503, -1.786, 0.313, -0.302, 0.768, -0.947, 1.487, 0.075}
$W_{1,2}$ = {-0.256, -0.026, 1.874, 1.488, -0.8, -0.804, 0.039, -0.957, 0.462, 1.677}

Suppose that the first gene was chosen to recombine and point 1 (between positions 0 and 1) and point 12 (between positions 11 and 12) were chosen as recombination points. Then the following offspring is formed:



a.

01234567890123456**7890123456789012**3456
DTQaababaabbaabba**0571745736284668**2867-[m]

W_m = {-1.64, -1.834, -0.295, 1.205, -0.807, 0.856, 1.702, -1.026, -0.417, -1.061}
T_m = {-1.14, 1.177, -1.179, -0.74, 0.393, 1.135, -0.625, 1.643, -0.029, -1.639}

b.

01234567890123456**7890123456789012**3456
DTQaababaabbaabba**0571846687457362**2867-[d]

W_d = {-1.64, -1.834, -0.295, 1.205, -0.807, 0.856, 1.702, -1.026, -0.417, -1.061}
T_d = {-1.14, 1.177, -1.179, -0.74, 0.393, 1.135, -0.625, 1.643, -0.029, -1.639}

Fig. 1. Testing new combinations of existing weights and thresholds by domain-specific transposition. **a)** The mother neural network. **b)** The daughter neural network created by domain-specific transposition. Note that the network architecture is the same for both mother and daughter and that $W_m = W_d$ and $T_m = T_d$. However, mother and daughter are different because different combinations of weights and thresholds are expressed in these individuals.

01234567890123456012345678901234 56
Taabbbabb978**93255QDbabbabb96369304-**[0]
Q**Tababaab143**72192QDbabbaaa81327963-[1]

with the weights encoded in exactly the same arrays as the parents. However, due to recombination, the weights expressed in the parents are different from those expressed in the offspring (compare their expressions in Figure 2).

a.

0123456789012345601234567890123456
TTababaab14393255QDbabbabb96369304-[0]
Qaabbbabb97872192QDbabbaaa81327963-[1]

⇓

0123456789012345601234567890123456
Taabbbabb978**93255QDbabbabb96369304**-[0]
Q**Tababaab143**72192QDbabbaaa81327963-[1]

$W_{0,1}$ = {-0.78, -0.521, -1.224, 1.891, 0.554, 1.237, -0.444, 0.472, 1.012, 0.679}
$W_{0,2}$ = {-1.553, 1.425, -1.606, -0.487, 1.255, -0.253, -1.91, 1.427, -0.103, -1.625}

$W_{1,1}$ = {-0.148, 1.83, -0.503, -1.786, 0.313, -0.302, 0.768, -0.947, 1.487, 0.075}
$W_{1,2}$ = {-0.256, -0.026, 1.874, 1.488, -0.8, -0.804, 0.039, -0.957, 0.462, 1.677}

b.

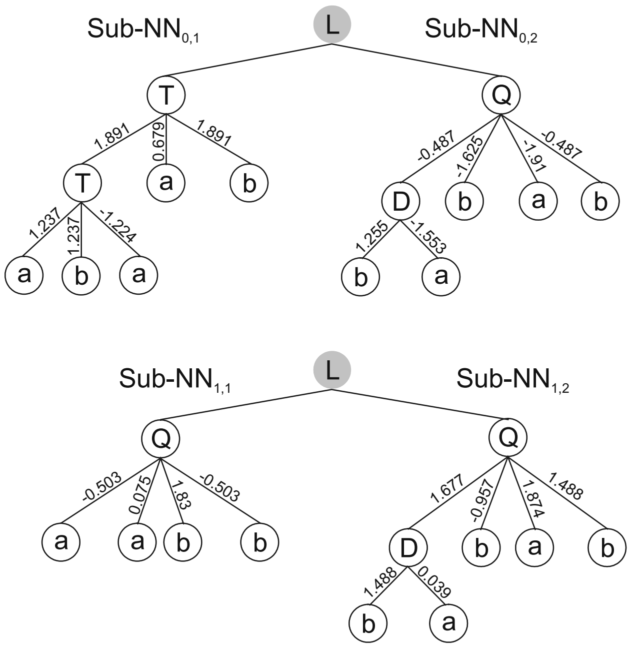


Fig. 2. Intragenic two-point recombination in multigenic chromosomes encoding neural networks. **a)** An event of intragenic two-point recombination between two parent chromosomes resulting in two new daughter chromosomes. Note that the set of weights is not modified by recombination. **b)** The sub-NNs codified by the parent chromosomes. **c)** The sub-NNs codified by the daughter chromosomes. Note that the sub-NNs encoded in the second gene are not modified. "L" represents a generic linking function.

It is worth emphasizing that this gene-restricted two-point recombination allows a greater control of the recombination effects and, consequently, permits a finer tuning of evolution. If we were to use one-point and two-point recombination as used in the basic gene expression algorithm, i.e., disrupting chromosomes anywhere, the fine adjustment of the weights would be an almost impossible task. Restricting two-point recombination to only one gene, however, ensures that only this gene is modified and, consequently, the weights and thresholds of the remaining genes are kept in place.

Remember, however, that intragenic two-point recombination is not the only source of recombination in multigenic neural nets: gene recombination is fully operational in these systems and it can be combined with gene transposition to propel evolution further. And in unigenic systems, the standard one-point and two-point recombination are also fully operational as only one gene is involved.

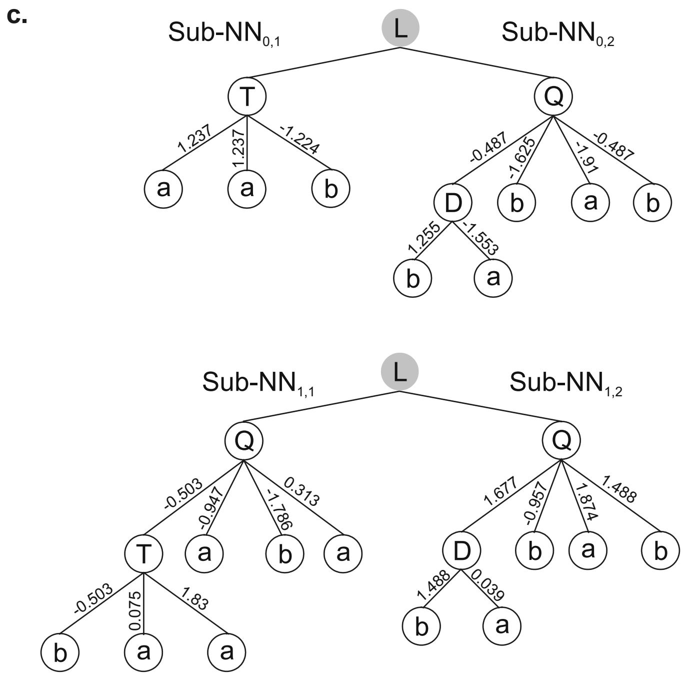


Fig. 3. *Continued.*

Direct Mutation of Weights and Thresholds

In the previous sub-sections it was shown that all genetic operators contribute directly or indirectly to move the weights and thresholds around. And, in fact, this constant shuffling of weights and thresholds is more than sufficient to allow an efficient evolution of GEP-nets as long as an appropriate number of weights and thresholds is randomly created at the beginning of each run. However, special mutation operators that replace the value of a particular weight or threshold by another can also be easily implement (see Figure 3).

This operator randomly selects particular targets in the arrays in which the weights or thresholds are kept, and randomly generates a new real-valued number. Consider for instance the array:

$W_{i,j}$ = {-0.433, -1.823, 1.255, 0.028, -1.755, -0.036, -0.128, **-1.163**, 1.806, 0.083}

encoding the weights of gene *j* on chromosome *i*. Suppose a mutation occurred at position 7, changing the weight -1.163 occupying that position into -0.494, obtaining:

$W_{i,j}$ = {-0.433, -1.823, 1.255, 0.028, -1.755, -0.036, -0.128, **-0.494**, 1.806, 0.083}

The consequences of this kind of mutation are very diverse: they might be neutral in effect (for instance, when the gene itself is neutral or when the weight/threshold has no expression on the sub-neural net) or they might have manifold effects (for instance, if the weight/threshold modified happened to be used more than once in the expression of the sub-NN as shown in Figure 3).

Interestingly, this kind of mutation seems to have a very limited importance and better results are obtained when this operator is switched off. Indeed, the direct mutation of numerical constants in function finding problems produces identical results (Ferreira 2003). Therefore, we can conclude that a well dimensioned initial diversity of constants, be they numerical constants of a mathematical expression or weights/thresholds of a neural net, is more than sufficient to allow their evolutionary tuning. In all the problems presented in this work, a set of 10 weights W = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} was used.

a.

```
0123456789012345678901234567890012
TDQDbaabababaaaaa7986582527723251-[m]
```

W_m = {**-0.202**, -1.934, **-0.17**, 0.013, 1.905, 1.167, 1.801, -1.719, 1.412, 0.434}

⇓

W_d = {**1.49**, -1.934, **1.064**, 0.013, 1.905, 1.167, 1.801, -1.719, 1.412, 0.434}

```
0123456789012345678901234567890012
TDQDbaabababaaaaa7986582527723251-[d]
```

b.

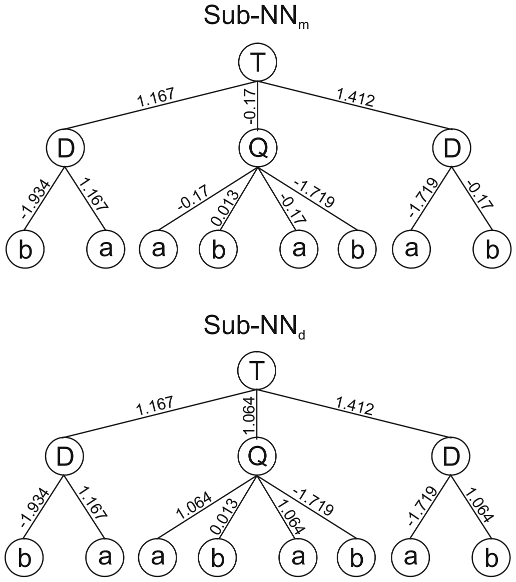


Fig. 3. Illustration of direct mutation of weights. **a)** The mother and daughter chromosomes with their respective weights. In this case, weights at positions 0 and 2 were mutated. Note that the mother and daughter chromosomes are the same. **b)** The mother and daughter neural nets encoded in the chromosomes. Note that the point mutation at position 2 (-0.17) has manifold effects as this weight appears four times in the neural network. Note also that the mutation at position 0 is an example of a neutral mutation as it has no expression on the neural net (indeed, mutations at positions 4, 6, and 9 would also be neutral).

Solving Problems with NNs Designed by GEP

The problems chosen to illustrate the evolution of linearly encoded neural networks are two well-known problems of logic synthesis. The first, the exclusive-or problem, was chosen both for its historical importance in the neural network field and for its simplicity, allowing an easy understanding of the evolved neural networks. The second, the 6-bit multiplexer, is a rather complex problem and can be useful for evaluating the efficiency of this new algorithm.

Neural Network for the Exclusive-or Problem

The XOR is a simple Boolean function of two activities and, therefore, can be easily solved using linearly encoded neural networks. Its rule table is shown in Table 1.

Table 1. Lookup table for the exclusive-or function.

a	b	o
0	0	0
0	1	1
1	0	1
1	1	0

The functions used to solve this problem have connectivities 2, 3, and 4, and are represented, respectively, by “D”, “T”, and “Q”, thus the function set F = {D, T, Q}; the terminal set T = {a, b}; and the set of weights W = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} with values randomly chosen from the interval [-2, 2]. For the experiment summarized in the first column of Table 2, an *h* = 4 was chosen and, therefore, hundreds of different correct solutions to the XOR function were found. Most of them are more complicated than the conventional solution (2.1) shown above which uses seven nodes; others have the same degree of complexity evaluated in terms of total nodes; but, as will next be shown, other solutions are surprisingly more parsimonious than the conventional solution mentioned above.

The first solution found in run 0 of the experiment summarized in the first column of Table 2 is shown below:

```
012345678901234567890123456789012
TQaTaaababbbabaaa6085977238275036
```

W = {1.175, 0.315, -0.738, 1.694, -1.215, 1.956, -0.342, 1.088, -1.694, 1.288}

Its expression is shown in Figure 4. It is a rather complicated solution to the XOR function, but remember that evolutionary algorithms thrive in slightly redundant architectures (Ferreira 2002) and, as shown in Table 2, the success rate for this problem using this non-compact chromosomal organization is higher (77%) than the obtained with more compact organizations with h = 2 (30%).

a.

```
0123456789012345678901234567890l2
TQaTaaababbbabaaa6085977238275036
```

W = {1.175, 0.315, -0.738, 1.694, -1.215, 1.956, -0.342, 1.088, -1.694, 1.288}

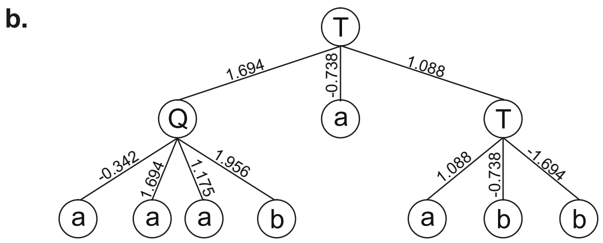


Fig. 4. A perfect, slightly complicated solution to the exclusive-or problem evolved with GEP-NNs. **a)** Its chromosome and respective weights. **b)** The fully expressed neural network encoded in the chromosome.

However, GEP can be useful to search for parsimonious solutions, and a very interesting parsimonious solution to the XOR function was found in another experiment. The parameters used per run in this experiment are summarized on the second column of Table 2. It is worth emphasizing that the compact organization with h = 2 was chosen in order to search for more parsimonious solutions than the canonical solution to the XOR function. One such solution is shown below:

```
01234567890123456
TDbabaabb88399837
```

W = {0.713, -0.774, -0.221, 0.773, -0.789, 1.792, -1.77, 0.443, -1.924, 1.161}

which is a perfect, extremely parsimonious solution to the XOR problem. Its full expression is shown in Figure 5. Indeed, several perfect solutions with this kind of structure were found in this experiment.

Table 2. Parameters for the exclusive-or problem.

	Redundant System	Compact System
Number of runs	100	100
Number of generations	50	50
Population size	30	30
Number of fitness cases	4	4
Function set	D T Q	D T Q
Terminal set	a b	a b
Weights array length	10	10
Weights range	[-2, 2]	[-2, 2]
Head length	4	2
Number of genes	1	1
Chromosome length	33	17
Mutation rate	0.061	0.118
One-point recombination rate	0.7	0.7
IS transposition rate	0.1	--
IS elements length	1	--
RIS transposition rate	0.1	--
RIS elements length	1	--
Dw-specific transposition rate	0.1	0.1
Dw-specific IS elements length	2,3,5	2,3,5
Success rate	77%	30%

a.

```
01234567890123456
TDbabaabb88399837
```

W = {0.713, -0.774, -0.221, 0.773, -0.789, 1.792, -1.77, 0.443, -1.924, 1.161}

b.

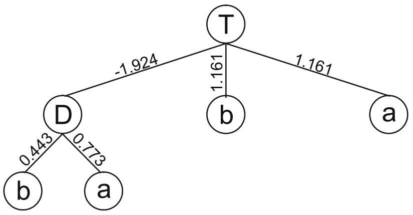


Fig. 5. A perfect, extremely parsimonious solution to the exclusive-or problem discovered with GEP designed neural networks. **a)** Its chromosome and corresponding array of weights. **b)** The fully expressed neural network encoded in the chromosome.

Neural Network for the 6-Multiplexer

The 6-bit multiplexer is a complex Boolean function of six activities. Its rule table is shown in Table 3.

Table 3. Lookup table for the 6-multiplexer. The output bits are given in lexicographic order starting with 000000 and finishing with 111111.

00000000	11111111	00001111	00001111
00110011	00110011	01010101	01010101

For this problem, and in order to simplify the analysis, a rather compact chromosomal organization was chosen and the "Q" function was not included in the function set. Thus, F = {3U, 3D, 3T}, where "U" represents a function with connectivity one; T = {a, b, c, d, e, f}, representing the six arguments to the 6-multiplexer function; and W = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, each taking values from the interval [-2, 2].

For the experiment summarized in the first column of Table 4, single-gene chromosomes were chosen so that the simulation of the 6-multiplexer function, a four modular function, went totally unbiased. One of the most parsimonious solutions designed by GEP-nets is shown in Figure 6.

Obviously, we could explore the multigenic nature of GEP chromosomes and evolve multigenic neural networks. The solutions found are, however, structurally more constrained as we have to choose some kind of linking function (Ferreira 2001) to link the sub-neural nets encoded by each gene. For this problem, the Boolean function OR was chosen to link the sub-NNs. (If the mixing of OR with "U", "D", and "T" functions is confusing, think of OR as a function with connectivity two with a threshold and weights all equal to 1, and you have a neural net for the OR function.)

In the experiment summarized in the second column of Table 4, four genes posttranslationally linked by OR were used. The first solution found in this experiment is shown in Figure 7. Note that some weights in genes 1 and 2 have identical values, and that the same happens for genes 3 and 4. This most probably means that these genes share a common ancestor.

Table 4. Parameters for the 6-multiplexer problem.

	Unigenic System	Multigenic System
Number of runs	100	100
Number of generations	2000	2000
Population size	50	50
Number of fitness cases	64 (Table 3)	64 (Table 3)
Function set	3U 3D 3T	3U 3D 3T
Terminal set	a b c d e f	a b c d e f
Linking function	--	O
Weights array length	10	10
Weights range	[-2, 2]	[-2, 2]
Head length	17	5
Number of genes	1	4
Chromosome length	103	124
Mutation rate	0.044	0.044
Intragenic two-point recombination rate	0.6	0.6
Gene recombination rate	--	0.1
Gene transposition rate	--	0.1
IS transposition rate	0.1	0.1
IS elements length	1,2,3	1,2,3
RIS transposition rate	0.1	0.1
RIS elements length	1,2,3	1,2,3
Weights mutation rate	0.002	0.002
Dw-specific transposition rate	0.1	0.1
Dw-specific IS elements length	2,3,5	2,3,5
Success rate	4%	6%

a)

```
TbDTTTfTTaUDcUUTTafeefebabbdabffddfcfeeeabcabfabdcfe...
   ...709761631479459597193997465381760511137453583952159
```

W = {0.241, 1.432, 1.705, -1.95, 1.19, 1.344, 0.925, -0.163, -1.531, 1.423}

b)

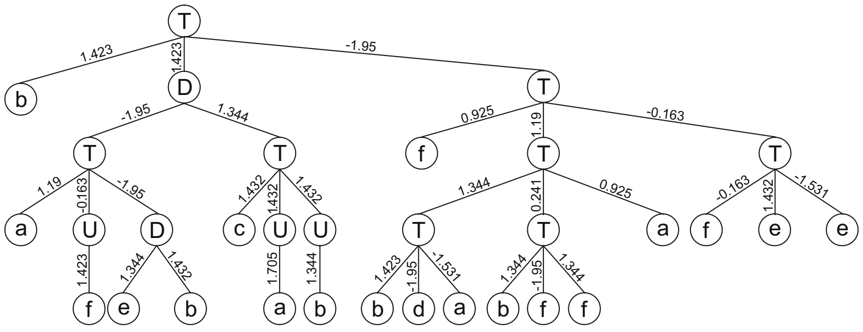


Fig. 6. A perfect solution to the 6-multiplexer function discovered with GEP-nets. **a)** Its chromosome and corresponding array of weights. **b)** The fully expressed neural network encoded in the chromosome.

a.

```
0123456789012345678901234567890
TecTDdfafabdddfa487674791701403-[1]
TDcbTbadddfceacc501702156029560-[2]
TfTTUbadbcdffdce593993321226318-[3]
TDTbaceaaeeacacd072636270049968-[4]
```

W_1 = {1.126, 0.042, 1.588, -0.03, -1.91, 1.83, -0.412, 0.607, -0.294, -0.659}
W_2 = {-1.961, 1.161, 1.588, -0.03, -1.91, 1.762, -0.412, -0.121, -0.294, -0.659}
W_3 = {1.558, -0.69, 0.921, 0.134, 0.468, -1.534, 0.966, 1.399, 0.023, 0.915}
W_4 = {1.558, 0.767, 0.076, 0.071, 0.468, -1.534, 1.387, -1.857, -1.88, 0.331}

b.

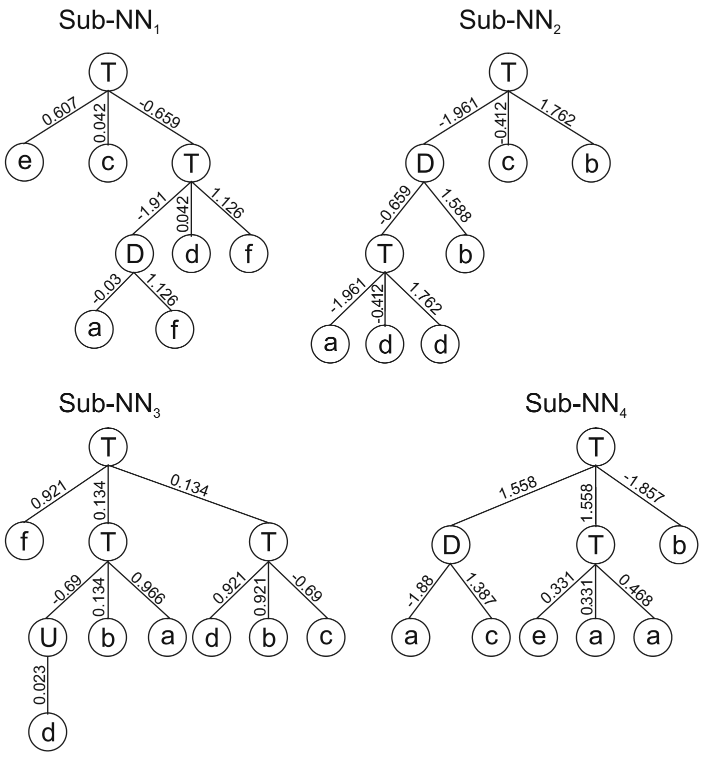


Fig. 7. A perfect solution to the 6-multiplexer problem encoded in a four-genic chromosome. **a)** Its chromosome with each gene shown separately. W_1-W_4 are the arrays containing the weights of each gene. **b)** The sub-neural networks codified by each gene. In this perfect solution, the sub-NNs are linked by OR.

Conclusions

The new algorithm presented in this work allows the complete induction of neural networks encoded in linear chromosomes of fixed length (the genotype) which, nonetheless, allow the evolution of neural networks of different sizes and shapes (the phenotype). Both the chromosomal organization and the genetic operators especially developed to evolve neural networks allow an unconstrained search throughout the solution space as any modification made in the genotype always results in valid phenotypes. Furthermore, as shown for the 6-multiplexer problem presented in this work, the multigenic nature of GEP-nets can be further explored to evolve complex neural networks with multiple outputs.

References

Anderson, J. A. (1995), *An Introduction to Neural Networks*, MIT Press.

Angeline, P. J., G. M. Saunders, and J. B. Pollack (1993). "An evolutionary algorithm that constructs recurrent neural networks," *IEEE Transactions on Neural Networks*, 5: 54-65.

Braun, H. and J. Weisbrod (1993), "Evolving feedforward neural networks," In *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms*, Innsbruck, Springer-Verlag.

Dasgupta, D. and D. McGregor (1992), "Designing application-specific neural networks using the structured genetic algorithm," In *Proceedings of the International Conference on Combinations of Genetic Algorithms and Artificial Neural Networks*, pp. 87-96.

Ferreira, C. (2001), "Gene expression programming: A new adaptive algorithm for solving problems," *Complex Systems*, 13 (2): 87-129.

Ferreira, C. (2002), "Genetic representation and genetic neutrality in gene expression programming," *Advances in Complex Systems*, 5 (4): 389-408.

Ferreira, C. (2003), "Function finding and the creation of numerical constants in gene expression programming," In J. M. Benitez, O. Cordon, F. Hoffmann, and R. Roy, eds, *Advances in Soft Computing: Engineering Design and Manufacturing*, pp. 257-266, Springer-Verlag.

Gruau, F., D. Whitley, and L. Pyeatt (1996), "A comparison between cellular encoding and direct encoding for genetic neural networks," In J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, eds, *Genetic Programming: Proceedings of the First Annual Conference*, pp. 81-89, Cambridge, MA, MIT Press.

Koza, J. R. and J. P. Rice (1991), "Genetic generation of both the weights and architecture for a neural network," In *Proceedings of the International Joint Conference on Neural Networks*, Volume II, IEEE Press.

Lee, C.-H. and J.-H. Kim (1996), "Evolutionary ordered neural network with a linked-list encoding scheme," In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pp. 665-669.

Mandischer, M. (1993), "Representation and evolution of neural networks," In R. F. Albrecht, C. R. Reeves, and U. C. Steele, eds, *Artificial Neural Nets and Genetic Algorithms*, pp. 643-649, Springer Verlag.

Maniezzo, V. (1994), "Genetic evolution of the topology and weight distribution of neural networks," *IEEE Transactions on Neural Networks*, 5 (1): 39-53.

Opitz, D. W. and J. W. Shavlik (1997), "Connectionist theory refinement: Genetically searching the space of network topologies," *Journal of Artificial Intelligence Research*, 6: 177-209.

Pujol, J. C. F. and R. Poli (1998), "Evolving the topology and the weights of neural networks using a dual representation," *Applied Intelligence Journal, Special Issue on Evolutionary Learning*, 8(1): 73-84.

Yao, X. and Y. Liu (1996), "Towards designing artificial neural networks by evolution," *Applied Mathematics and Computation*, 91(1): 83-90.

Zhang, B.-T. and H. Muhlenbein (1993), "Evolving optimal neural networks using genetic algorithms with Occam's razor," *Complex Systems*, 7: 199-220.

Particle Swarm Optimisation from lbest to gbest

Hongbo Liu, Bo Li, Ye Ji, and Tong Sun

Dept. of Computer, Dalian University of Technology, Dalian, 116023, China
abrahamliu09@yahoo.com

The effects of various neighborhood models on the particle swarm algorithm were investigated in this paper. We also gave some additional insight into the PSO neighborhood model selection topic. Our experiment results testified that the gbest model converges quickly on problem solutions but has a weakness for becoming trapped in local optima, while the lbest model converges slowly on problem solutions but is able to "flow around" local optima, as the individuals explore different regions. The gbest model is recommended strongly for unimodal objective functions, while a variable neighborhood model is recommended for multimodal objective functions.

1 Introduction

Particle swarm optimization (PSO) was originally introduced by J. Kennedy et al. in 1995 as an optimisation technique inspired by swarm intelligence and theory in general such as bird flocking, fish schooling and even human social behavior [1]. Furthermore, the whole idea and structure of the algorithm is inspired by evolutionary computation. Later PSO has turned out to be a worthy alternative to the standard genetic algorithm and other iterative optimisation techniques [2, 3]. Particle swarm optimizer has exhibited good performance across a wide range of applications [4, 5, 6, 7]. The PSO algorithm is initialized with a population of random candidate solutions, conceptualized as particles. Each particle is assigned a randomized velocity and is iteratively moved through the problem space. It is attracted towards the location of the best fitness achieved so far by the particle itself and by the location of the best fitness achieved so far across the whole population (global version of the algorithm). The two most commonly used methods are known as gbest model and lbest model in particle swarm optimization. The gbest model converges quickly on problem solutions but has a weakness for becoming trapped in local optima, while lbest model is able to "flow around" local optima, as the individuals explore different regions [8]. The lore is based on experience and some data, but population topologies have not been systematically explored. Sugarthan [9], Peer [10] researched effects of neighborhood models in different perspective, but some divergence existed among those results. And Kennedy and his colleague's research manipulated some sociometric variables that are hypothesized to affect performance. They discovered that previous assumptions may not have been correct [11]. The effects of the two neighborhood models on the particle swarm algorithm are systematically investigated in this paper. The present work gives some additional insight into the PSO neighborhood model selection topic.

2 Particle Swarm Optimization Algorithm

The basic PSO model consists of a swarm of particles moving in an n-dimensional search space where a certain quality measure, the fitness, can be calculated. Each particle has a position represented by a position-vector $\overrightarrow{x}$ and a velocity represented by a velocity-vector $\overrightarrow{v}$. Each particle remembers its own best position so far in a vector $\overrightarrow{p}_i$, i is the index of the particle and the dth dimensional value of the vector $\overrightarrow{p}_i$ is p_{id}. Further, a neighborhood relation is defined for the swarm. The best position-vector among all the neighbors of a particle is then stored in the particle as a vector $\overrightarrow{p}_g$ and the dth dimensional value of the vector $\overrightarrow{p}_g$ is p_{gd}. At each iteration time t, the velocity is updated and the particle is moved to a new position. Firstly the update of the velocity from the previous velocity to the new velocity is determined by Equ.(1). r_i (i=1, 2) is the random numbers, which range are in [-1, +1]. c_1 is the coefficient of the recognition component and c_2 is the coefficient of the social component. w is the inertia factor. And then the new position is determined by the sum of the previous position and the new velocity by Equ.(2), a neighborhood relation is defined for the swarm:

$$v_{id}(t+1) = w * v_{id}(t) + c_1 * r_1 * (p_{id}(t) - x_{id}(t)) + c_2 * r_2 * (p_{gd}(t) - x_{id}(t)). \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1). \quad (2)$$

In the PSO algorithm, a particle decides where to move next, considering its own experience, which is the memory of its best past position, and the experience of its most successful neighbor. At each iteration, the particle with the best fitness in the local neighborhood, designated g, and the current particle are combined to adjust the velocity along each dimension, and that velocity is then used to compute a new position for the particle. The portion of the adjustment to the velocity influenced by the individual's previous best position is considered the cognition component, and the portion influenced by the best in the neighborhood is the social component.

3 gbest Model vs. lbest Model

The particle swarm algorithm can be described generally as a population of vectors whose trajectories oscillate around a region which is defined by each individual's previous best success and the success of some other particle. Various methods have been used to identify some other particle to influence the individual. Eberhart and Kennedy called the two basic methods as "gbest model" and "lbest model" [8]. In the lbest model, particles have information only of their own and their nearest array neighbors' best (lbest), rather than that of the entire group. Namely, in Equ.(1), gbest is replaced by lbest in the model. So a new neighborhood relation is defined for the swarm:

$$v_{id}(t+1) = w * v_{id}(t) + c_1 * r_1 * (p_{id}(t) - x_{id}(t)) + c_2 * r_2 * (p_{ld}(t) - x_{id}(t)). \quad (3)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1). \quad (4)$$

In the gbest model, the trajectory of each particle's search is influenced by the best point found by any member of the entire population. The best point/particle acts as an attractor, pulling all the particles towards it. Eventually all particles will converge to this position. The lbest model allows each individual to be influenced by some smaller number of adjacent members of the population array. The particles selected to be in one subset of the swarm have no direct relationship to the other particles in the other neighborhood. Typically lbest neighborhoods comprise exactly two neighbors. When the number of neighbors increases to all but itself in the lbest model, the case is equivalent to the gbest model.

There may be different concepts for neighborhood, it can be seen as spatial neighborhood where it is determined by the Euclidean distance between the positions of two particles, or as a sociometric neighborhood (e.g.: the index position in the storing array). They both have social background [12, 13]. The latter is the most commonly used for two main motives: if space coordinates were to represent mental abilities or skills, two very similar individuals may never come to meet in their lifetime, as to elements of the same family, which may differ significantly from each other, but still, they will always be neighbors. The other motive is related with the computational effort required to process the Euclidean distance, when faced with large number of particles or dimensions. At each iteration, the distance between every two particles would have to be calculated and for each particle the nearest k neighbors would have to be sorted out.

4 Algorithm Description from lbest to gbest

In the PSO algorithm, particles are directed to search global optimum in the problem's solution space. For lbest model and gbest model, the main difference is neighborhood size. The lbest model and gbest model would be attained while we use variable neighborhood size from 2 to (*population size*-1). The main pseudo-code for particle-searching is as follows:

```
Initialize $k$, $c_1$, $c_2$, and all other parameters
Initialize the individual's position and velocity
Do
    $w = 0.4 + \frac{IterationMaxNum - IterationNum}{2IterationMaxNum}$
    Calculate the fitness value of each particle
    $\overrightarrow{p}_g = \overrightarrow{p}_i$
    For $i$ to Population Size
        if $f(\overrightarrow{p}_i) > f(\overrightarrow{x}_i)$ then $\overrightarrow{p}_i = \overrightarrow{x}_i$
        Find $k$ nearest neighbors
        $\overrightarrow{p}_l = min(\overrightarrow{p}_{neighbours})$
        For $d = 1$ to dimension
            Update each dimension value of $\overrightarrow{p}_i$
        Next $d$
    Next $i$
While $(f(\overrightarrow{p}_g > Minerror))$ or $(IterationNum < IterationMaxNum)$
```

There are two major measures of performance in an optimizer such as the particle swarm [11]. The first is best function result attained after some number of iterations. It is possible however for the algorithm to rapidly attain a relatively good result while becoming trapped on a local optimum. Thus a second dependent measure is the number of iterations required for the algorithm to meet a criterion. If it doesn't meet by a large iteration time, the measure is considered infinite, that is, it is reported as if the criterion would never be met. So a third dependent measure is derived from the second. That is a simple binary variable describing whether the version attains the criterion or not. The success rate would be obtained from the average and median iterations for successful runs.

5 Benchmark Test Functions

Several types of particle swarms were used to optimize a set of unconstrained real-valued benchmark function. For each of these functions, the goal is to find the global minimiser. Stated formally:
Given $f : R^n \rightarrow R$, find $x^* \in R^n$ for which $f(x^*) \leq f(x), \forall x \in R^n$.

(1)Rosenbrock variant (De Jong's f2)

$$f(x) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2 \tag{5}$$

where $-2.048 \leq x_i \leq 2.048$, continuous, unimodal; $f(x^*) = 0$ with $x^* = (1, 1)$.

(2)Griewank's function

$$f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}}) + 1 \qquad (6)$$

where $-300 \leq x_i \leq 300$, continuous, multimodal; $f(x^*) = 0$ with $x^* = 0$.

6 Experiment Setting

The performance of PSO is correlative directly with the parameters selection. For obtaining a good performance in our experiments, all but the neighborhood size are used following the recommended setting in [14]. The inertia weight is implemented as a linearly decreasing function, initially set to 0.9, decreasing to 0.4 over the first 1500 iterations if the iterations are above 1500, and remaining at 0.4 over the remainder of the run. In order to analyze the influence of lbest model to gbest model, we set a smaller value for c_1 and a larger value for c_2 in our most of experiments. Incorporating Clerc's constriction factor [15], c_2, the coefficient of the social component, is set from 1.0 to 2.4, while c_1, the coefficient of the recognition component, is set from 1.5 to 2.0. V_{max} sets equal to X_{max}, and in every experiment, each function was run for 10 repetitions. We use a variant of the Rosenbrock's benchmark function with $n = 2$, and 30 dimensions for the Griewank's function in our experiment. The success rate was also taken into account. The population size is set to 21.

7 Results and Discussions

In Figure 1 and 2, the results of iteration procedures display for the two benchmark functions when the neighborhood size is varying. Most of iteration procedures were at standstill state; all particles had got together at one position. But some of iteration procedures still went on, many particles were at different positions dispersedly. The runs were ended because the iteration number partial criterion was met. In our work they misleaded us in our earlier analysis phase, because the values in lbest model are always larger than that in gbest model. When the fitness values between different groups were compared separately, we could draw a rational conclusion: In general, the gbest model converges quickly on problem solutions but has a weakness for becoming trapped in local optima, while lbest model converges slowly but is able to "flow around" local optima. No surprisingly, the good fitness value appears to depend on the form of the objective function.

We use the sketch map to illuminate the deference between lbest model and gbest model, Figure 3 for unimodal objective function and Figure 4 for multimodal objective function, respectively. In Figure 3, the fitness value in lbest model is larger than that in gbest model at I1 and I3. If the iteration

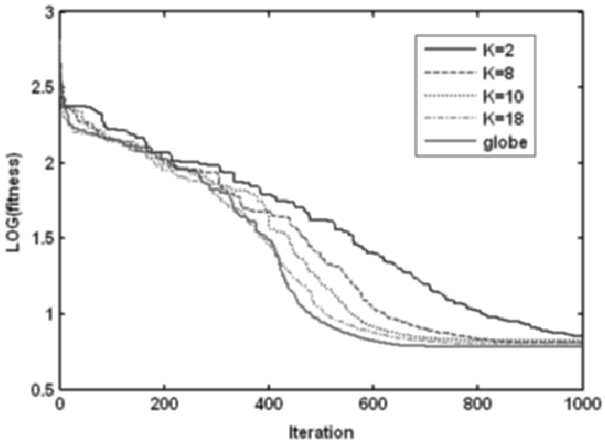


Fig. 1. Iteration procedures in PSO for Rosenbrock variant (De Jong's f2)

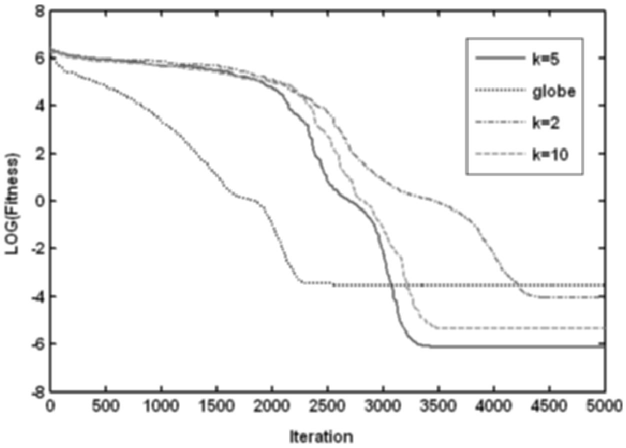


Fig. 2. Iteration procedures in PSO for Griewank's function

number partial criterion in the PSO algorithm is met in the run, we would get a larger fitness value in lbest model than that in gbest model. Iteration number in lbest model is more than that in gbest model at Goalf1 and Goalf2. So the minimum error partial criterion in the PSO algorithm is met in the run, we need use more iteration time in lbest model than that in gbest model.

As Figure 4 indicates the state in lbest model and the gbest model for multimodal objective functions, it is possible for both lbest model and gbest model to arrive local optima due to premature convergence. The gbest model converges quickly but there is a larger probability to trap in local optima, while lbest model spends so much time to explore different regions that it converges very slowly but there is a larger probability to arrival the global optima. In

Figure 4, the fitness value in lbest model is larger than that in gbest model at a small iteration time I1. In other words, when we set a small iteration time as the criterion in the PSO algorithm in order to that the iteration time partial criterion is met in the run, we get a larger fitness value in lbest model than that in gbest model. But the results would be reversed possibly at I2 and I3, which are more iteration time. When we set larger goal fitness values as the criterion in the PSO algorithm in order to that the iteration number partial criterion is met in the run, we need a more iteration number in lbest model than that in gbest model. But if we set larger goal fitness values as the criterion in the PSO algorithm and there is no other partial criterion, the success rates would be much more than that in gbest model possibly.

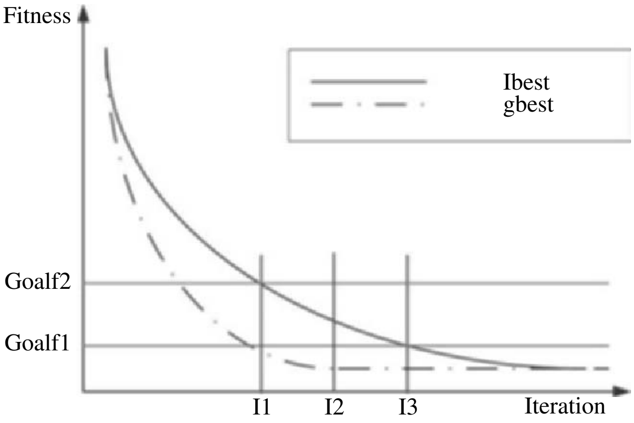


Fig. 3. PSO: from lbest to gbest model for unimodal function

For unimodal objective functions, gbest model is recommended strongly, because the best fitness values of gbest model are almost similar to that of lbest model, and gbest model converges much quickly. For multimodal objective functions, the user can take well-informed decisions according to the desired exploration-exploitation trade-off: either favour exploration by a thorough sampling of the solution space for a robust location of the global optimum at the expense of a large number of objective function evaluations or, on the contrary, favour exploitation resulting in a quick convergence but to a possibly non-optimal solution. If PSO would be used to attain better fitness values for multimodal functions, we recommend a variable neighborhood model, "vbest model", to inherit both quick convergence in gbest model and "flow around" local optima in lbest mode. At the beginning procedure, the number of neighbors, k, is 2, and increasing gradually to (*swarm population size*-1) while iteration number is increasing. The fitness-iteration procedure of vbest model is similar to the omit curve (vbest) in Figure 4.

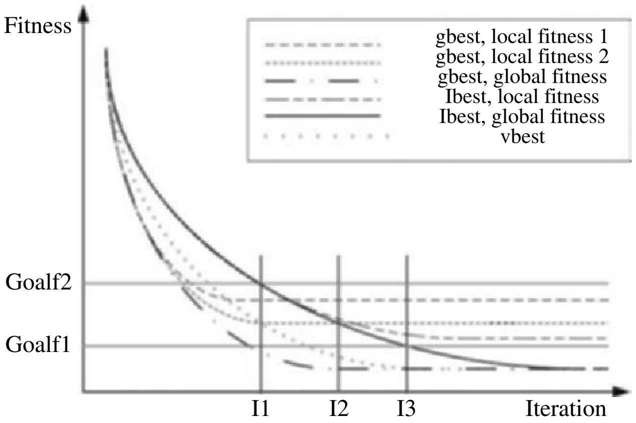


Fig. 4. PSO: from lbest to gbest model for multimodal function

8 Conclusion

The effects of various neighborhood models on the particle swarm algorithm had been investigated in this paper. We also gave some additional insight into the PSO neighborhood model selection topic. Our experiment results testified that gbest model converges quickly on problem solutions but has a weakness for becoming trapped in local optima, while lbest model converges slowly on problem solutions but is able to "flow around" local optima, as the individuals explore different regions. The gbest model is recommended strongly for unimodal objective functions, while a variable neighborhood model is recommended for multimodal objective functions.

Acknowledgments

The work is supported by NSFC (30170321, 90103033), MOE (KP0302) and MOST (2001CCA00700). The first author wishes to thank to Prof. Xiukun Wang, who listened patiently when he discussed some of his ideas with her, for her feedback and insight.The authors are grateful to the anonymous referees and Dr. Candida Ferreira for their constructive comments that improved the presentation of this work.

References

1. Kennedy J, Eberhart R C (1995) Particle swarm optimization. In: Proceedings of IEEE International Conference on neural networks, 1942-1948
2. Eberhart R C, Shi Y (1998) Comparison between genetic algorithms and particle swarm optimization. In: Proceedings of 7th annual conference on evolutionary computation, 611-616

3. Boeringer D W, Werner D H (2004) Particle swarm optimization versus genetic algorithms for phased array synthesis. IEEE Transactions on Antennas and Propagation, 52(3):771-779
4. Parsopoulos K E, Vrahatis M N (2002) Recent approaches to global optimization problems through particle swarm optimization. Natural Computing, 1:235-306
5. Sousa T, Silva A, Neves A (2003) A particle swarm data miner. In: Lecture Notes in Computer Science, Fernando M P, Salvador A (Eds.): EPIA 2003, LNAI 2902, Springer-Verlag, 43-53
6. Ting T, Rao M, Loo C K, Ngu S S (2003) Solving unit commitment problem using hybrid particle swarm optimization. Journal of Heuristics, 9:507-520
7. Chang B, Ratnaweera A, Halgamuge S (2004) Particle swarm optimisation for protein motif discovery. Genetic Programming and Evolvable Machines, 5:203-214
8. Kennedy J, Eberhart R (2001) Swarm Intelligence. Morgan Kaufmann Publishers, Inc., San Francisco, CA
9. Sugarthan P N (1998) Particle swarm optimiser with neighbourhood operator. In: Proceedings of IEEE International Conference on Evolutionary Computing, 3:1958-1964
10. Peer E S, van den Bergh F, Engelbrecht A P (2003) Using neighborhoods with the guaranteed convergence PSO. In: Proceeding of IEEE conference on Evolutionary Computation, 235-242
11. Kennedy J, Mendes R (2002) Population structure and particle swarm performance. In: Proceeding of IEEE conference on Evolutionary Computation, 1671-1676
12. Watts D J, Strogatz S H (1998) Collective dynamics of 'Small-World' networks. Nature, 393:40-442
13. Watts D J (1999) Small World: The dynamics of networks between order and randomness. Princeton University Press
14. Eberhart R C, Shi Y (2002) Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of IEEE International Congress on Evolutionary Computation, 84-88
15. Clerc M, Kennedy J (2002) The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation, 6(1):58-73

Multiobjective 0/1 Knapsack Problem using Adaptive ε -Dominance

Crina Groşan

Department of Computer Science
Babeş-Bolyai University, Kogălniceanu 1
Cluj-Napoca, 3400, Romania.
cgrosan@cs.ubbcluj.ro

Summary. The multiobjective 0/1 knapsack problem is a generalization of the well known 0/1 knapsack problem in which multiple knapsacks are considered. A new evolutionary algorithm for solving multiobjective 0/1 knapsack problem is proposed in this paper. This algorithm used a ε-dominance relation for direct comparison of two solutions. This algorithm try to improve another algorithm which also uses an ε domination relation between solutions. In this new algorithm the value of ε is adaptive (can be changed) depending on the solutions quality improvement. Several numerical experiments are performed using the best recent algorithms proposed for this problem. Experimental results clearly show that the proposed algorithm outperforms the existing evolutionary approaches for this problem.

1 Introduction

The knapsack problem is a well known combinatorial problem. It has been well studied in the single objective context. It is also called a master problem since many combinatorial problems can be formulated as a knapsack problem.

The 0/1 knapsack problem is a widely studied problem due to its practical importance. In the last years a generalization of this problem was well studied and many algorithms for solving this variant have been proposed. Of great interest are the evolutionary approaches for solving the multiobjective 0/1 knapsack problem. Many papers on the multiobjective knapsack problem and on the algorithms proposed for solving it can be found in the literature [1], [2], [3], [4], [5], [6], [7], [8], [10], [11], [13], [14], [15], [16], [19]. In this paper, we improve the performances of the ε Multiobjective Knapsack Algorithm (ε-MOKA) proposed in [5]. ε-MOKA uses the ε-dominance concept which is a generalization of the standard Pareto concept.

In section 2 of the paper both single and multiobjective 0/1 knapsack problems are presented. The description of the ε-MOKA technique is given in section 4 of the paper. The new proposed algorithm is presented in section 5. Some comparisons with the most recent algorithms (such as SPEA2, NSGA

II, PESA and ε-MOKA) are performed in section 6. A set of conclusions are given in section 7 of the paper.

2 Multiobjective 0/1 Knapsack Problem

The classical 0/1 knapsack problem can be formulated as follow: a set of n items and a knapsack of capacity c are considered. Each item has a profit p_j and a weight w_j. The problem is to select a subset of the items whose total weight does not exceed knapsack capacity c and whose total profit is maximum. Using the variables x_j (with x_j= 1 if the item j is selected and x_j= 0 otherwise) the problem can be written:

$$\text{maximize} \sum_{j=1}^{n} p_j x_j$$

$$\text{subject to} \sum_{j=1}^{n} w_j x_j \leq c$$

$$x_j \quad \in \{0,\ 1\},\ \mathrm{j} = \{1,\ \ldots,\ n\}.$$

The problem can be extended for an arbitrary number of knapsacks. The multiobjective 0/1 knapsack problem is defined as follows:

A set of n items and a set of k knapsacks are considered. For each item we know:

p_{ij} – profit of item j according to knapsack i;

w_{ij}– weight of item j according to knapsack i.

The capacity of each knapsack i is c_i.

The problem is to find a vector $x = (x_1,\ x_2,\ \ldots,\ x_n) \quad \in \{0,\ 1\}^n$ such that the capacity constraints

$$e_i(x) = \sum_{j=1}^{n} w_{ij} \cdot x_j \leq c_i, \quad (1 \leq i \leq k)$$

are satisfied and for which $f(x) = (f_1(x),\ f_2(x),\ldots,\ f_k(x))$ is maximum, where

$$f_i(x) = \sum_{j=1}^{n} p_{ij} \cdot x_j$$

and x_j= 1 if the item j is selected and x_j= 0 otherwise.

3 ε-Dominance

The definition of ε-Pareto dominance concept which is a generalization of the Pareto dominance concept is given below.

Definition

Consider a maximization problem. Let x, y be two decision vectors (solutions) from the search space. Solution x *ε-dominate* y if and only if the following conditions are fulfilled:

(i) $f_i(x) \geq f_i(y) + \varepsilon, \forall \quad i = 1, 2, \ldots, n,$
(ii)$\exists j \in \{1, 2, \ldots, n\}$: $f_j(x) > f_j(y) + \varepsilon$.

A similar definition of ε-dominance is given by Laumans et al. in [9]. In that definition ε is added at first condition from standard Pareto dominance definition only. Furthermore, in Laumanss definition the value of ε is fixed during the search process while in proposed approach this value is changed during the evolution process.

4 ε-MOKA Technique

Each ε-MOKA individual is a binary string. The value 1 for the position j of the chromosome means that the item j is selected to be included in a knapsack.

The algorithm uses steady-state as its underlying mechanism and can be described as follows:

The algorithm starts with a population of randomly generated individuals. For each chromosome the total items weight for each knapsack is computed. If there are knapsacks for which the allowed capacity is exceeded the items starting with the one for which the proportion utility/weight has the smaller value are eliminated. This process continues until there are no knapsacks for which the capacity is exceeded. All nondominated solutions are computed using the ε-dominance concept. The following steps are repeated until a termination condition is reached: Two nondominated solutions (the parents) are randomly chosen. The parents are recombined using uniform crossover operator and the offspring are mutated. For each offspring the procedure for eliminating items if the capacity of one of knapsacks is exceeded is used. The offspring enters the population and the dominated solutions are removed.

An individual is a set of selected items. Each knapsack is checked for overloading. If the total weight of the selected objects exceeds its capacity the items starting with the one for which the proportion utility/weight is lowest are eliminated.

5 Adaptive ε-MOKA

The difference between ε-MOKA and Adaptive ε-MOKA mainly consist in the way in which the value of ε is changed. In ε-MOKA technique ε value decreases by one after a given number of generations. This method seems do not be very efficient because for a bigger number of generations the initial value of ε have to be a great number or have to decrease rarely. In this paper we improve the performances of ε-MOKA by changing the way in which ε decrease. We try to adapt the ε value modification during the search process. Thereby, the value of ε decrease after a number of generations in which no improvement occurs in population. We denote this number by *Max_Worst_Iterations* and it is a parameter of the algorithm. After *Max_Worst_Iterations* in which no improvement occurs in solutions population the value of ε decreases with a given number (bigger or smaller depending on the initial value of epsilon and the test data: if we have many items this value can be bigger).

The reason behind this mechanism is to allow many solutions become non-dominated with each other. This may increase the diversity of parent solutions because non-dominated solutions are selected as parents. Another effect is that offspring are less likely to dominate other individuals in the current population.

The main steps of the new proposed technique are presented below:

Adaptive ε-MOKA technique description

```
begin
Initialize population;
For each individual from population set to 0 the value of Number_worst;
for nr = 1 to NumberOfGenerations do
    begin
      for i = 1 to popsize do
        for j = 1 to NumberOfKnapsacks do
          if exceed(pop[i], j)
          then eliminate(pop[i], j)
      Find nondominated solutions from population;
      for selection_number = 1 to popsize / 2 do
        begin
          Random select two individuals pop[ind1] and pop[ind2];
          Crossover(pop[ind1], pop[ind2]);
          Two offspring off1 and off2 are obtained;
          Mutate the offspring;
          for j = 1 to knapsacks_number do
           if exceed(off1, j)
           then eliminate(off1, j)
          for j = 1 to knapsacks_number do
           if exceed(off2, j)
           then eliminate(off2, j)
```

```
            if dominate(pop[ind1], off1)
            then Increase the value of Number_worst for pop[ind1];
            if Number_worst (for pop[ind1]) = Max_worst)
            then begin
              decrease the value of $\varepsilon$;
              set to 0 the value of Number_worst for pop[ind1]
            end
            (A similar test is applied for the second offspring)
            The offspring replace the worst individual from population if is
            better than this;
          end;
        end;
  end.
```

The procedure *exceed*$(pop[i], j)$ is used for each individual $pop[i]$ from population and for each knapsack j in order to determinate if there are knapsacks for which the capacity is exceeded. The procedure *eliminate*$(pop[i], j)$ is used in order to eliminate items from $pop[i]$ until the capacity of the knapsack j is not exceeded.

6 Experimental Results

We test our algorithm considering 750 items and two and three knapsacks. These test cases are two of the most difficult test data for knapsack problem. The results obtained by Adaptive ε-MOKA are compared to the results obtained by ε-MOKA, SPEA2, NSGA II and PESA. For this comparison C metric (presented below) is used.

6.1 C-Metric

The C metric was introduced by Zitzler [18]. Using C metric two sets of nondominated solutions can be compared to each other. The definition of C metric given in [18] is:

Definition. *(Coverage of two sets)*

Let X be the set of decision vectors for the considered problem and A, $B \subseteq X$ two sets of decision vectors. The function C maps the ordered pair (A, B) into the interval [0,1]:

$$C(A,\ B) = \frac{|\{b \in B\ /\ \exists a \in A{:}a \succeq b\}|}{|B|}.$$

Remarks

The value $C(A, B) = 1$ means that all decision vectors in B are dominated by A.

The value $C(A, B) = 0$ represent the situation when none of the points in B are dominated by A.

$C(A, B)$ is not necessary equal to 1 – $C(B, A)$.

6.2 Numerical Comparisons

In order to compare the results obtained by Adaptive ε-MOKA with the results obtained by ε-Moka, SPEA2, NSGA II and PESA the instances with 750 items and two and three knapsacks are considered [18]. These cases are two of the most difficult for the multiobjective knapsack problem.

General parameters of MOKA algorithm are presented in Table 1.

Table 1. Parameters used by Adaptiveε-MOKA, ε-MOKA, NSGA II, SPEA 2 and PESA.

Number of items	**Number of knapsacks**	**Population size**	**Number of function evaluations**
750	2	250	480,000
750	3	300	576,000

The value of ε for ε-MOKA is chosen as follows: at the beginning of the search process this value is a large one. In this way a low dominance is ensured and this allow us to preserve many solutions in the first generations. After a number of iterations the value of ε is decreased by 1.

The values of ε for ε-MOKA in all considered situations are presented in Table 2.

Table 2. The values of ε for ε-MOKA in the case of two and three knapsacks.

Number of knapsacks	**The value of ε**	**Number of generations after which ε becomes ε-1**
2	500	800
3	1500	1000

The value of ε for Adaptive ε-MOKA is 1000 and the value of *Max_Worst_Iterations* is 10.

Comparison of results

The results obtained in the case of two knapsacks by applying C metric for 480,000 functions evaluations are presented in Table 3.

Table 3. The results obtained for three knapsacks by Adaptive ε-*MOKA*, ε-MOKA, NSGA II, SPEA2 and PESA considering two knapsacks and 480,000 functions evaluations. Results are averaged over 30 runs.

	Adaptive ε-MOKA	**ε-MOKA**	**NSGA II**	**SPEA 2**	**PESA**
Adaptive ε-MOKA		1	1	1	1
ε-MOKA	0		0.9502	0.8693	0.98221
NSGA II	0	0		0.7016	0.6917
SPEA 2	0	0	0.2488		0.8413
PESA	0	0	0.2467	0.	0.1179

The values 0 on first column means that no solution from the final population obtained by Adaptive ε-MOKA is dominated by solutions from final populations obtained by NSGA II, SPEA 2 and PESA. The values 1 on the first line means all solutions from final populations obtained by ε-MOKA, NSGA II, SPEA 2 and PESA are dominated by solutions obtained by Adaptive ε-MOKA.

In test data considered in our experiments the knapsacks capacity are 20460 and 20351.5 respectively. Items utility varies between 30 and 100. In the picture below we compare the solutions from final population obtained by Adaptive ε-MOKA, ε-MOKA, NSGA II, PAES and SPEA 2. First objective (denoted by f_1) represents the total utility of selected item regarding first knapsack and the second objective (denoted f_2) represents the total utility of selected item regarding second knapsack. Bigger values of these objectives mean better results.

From this picture we can see results obtained by Adaptive ε-MOKA clearly dominate solutions obtained by the other considered algorithms.

The results obtained in the case of three knapsacks by applying C metric for 576,000 functions evaluations are presented in Table 4.

Analyzing these tables we can see the effectiveness of adapting ε modification during the search process. ε-MOKA to the multicriterion knapsack problem. By using the ε dominance technique at the beginning of search process many solutions (which otherwise could be eliminated from the evolution stage) have the opportunity to be improved. However, by decreasing the value of ε the standard Pareto dominance is obtained at the end of the search process. The uses of ε dominance may decrease the convergence speed comparing to the considered algorithms.

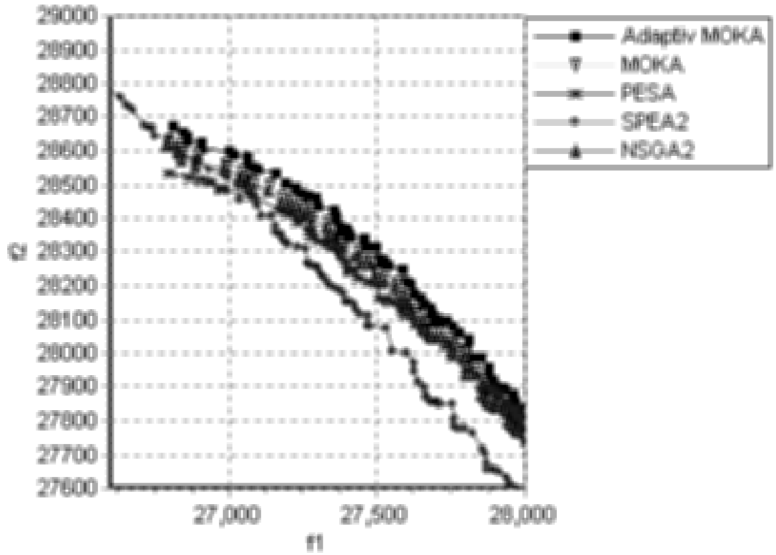


Fig. 1. Results obtained by Adaptive ε-MOKA algorithm, NSGA II, PAES and SPEA2.

Table 4. The results obtained for three knapsacks by ε-MOKA, NSGA II, SPEA2 and PESA considering 576,000 functions evaluations. Results are averaged over 30 runs.

	Adaptive ε-MOKA	**ε-MOKA**	**NSGA II**	**SPEA 2**	**PESA**
Adaptive ε-MOKA		1	1	1	1
ε-MOKA	0		0.999851	1	1
NSGA II	0	0.34914		0.99623	0.99170
SPEA 2	0	0.128210	0.832425		0.96905
PESA	0	0.128210	0.809001	0.939329	

7 Conclusions

In this paper a new evolutionary algorithm for solving 0/1 multiobjective knapsack problem has been proposed. The algorithm uses the ε – dominance concept which is a generalization of the Pareto dominance. The value of ε is not fixed. Larger values are considered for ε at the beginning of the search process. These values are decreased as the number of generations increase. This procedure proved to be very useful.

Several numerical experiments have been performed using several well-known instances of the knapsack problem. For these experiments two knapsacks and 100 items are considered. A comparison with SPEA, PESA and NSGA II is also performed. Experimental results have shown that the proposed algorithm significantly outperforms the compared algorithms in all experiments.

References

1. Balas, E., Zemel, E. An algorithms for large zero-one knapsack problems, in *Operations Research*, Vol 28, pp. 1130-1154, 1980.
2. Chvatal, V. Hard knapsack problems, Operations Research, Vol 28, pp. 1402-1411, 1980.
3. Corne, D.W., Knowles, J.D. The Pareto-Envelope based Selection Algorithm for Multiobjective Optimization, in Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature, Springer-Verlag, Berlin, 2000, pp. 839-848.
4. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA II, in M. S. et al. Eds, Parallel Problem Solving From Nature – PPSN VI, Springer-Verlag, Berlin, 2000, pp. 849 - 858.
5. Grosan, C. Improving the performance of evolutionary algorithms for the multiobjective 0/1 knapsack problem using ε -dominance, In Proceedings of Congress on Evolutionary Computation (CEC), Portland, 2004.
6. Ingargiola, G. P., Korsh, J. F. A reduction algorithm for zero-one single knapsack problems, in Management Science Vol. 20, pp. 460-463, 1975.
7. Jaszkiewicz, A. On the performance of Multiple Objective Local Search on the 0/1 Knapsack problem – A Comparative Experiment, IEEE Transaction on Evolutionary Computation, Vol 6, pp. 402-412, 2002.
8. Ko, I. Using AI techniques and learning to solve multi-level knapsack problems. PhD thesis, University of Colorado at Boulder, Boulder, CO, 1993.
9. Laumans, M., Thiele, L., Deb, K., and Zitzler, E.. Combining convergence and diversity in evolutionary multi-objective optimization. Evolutionary Computation 10(3), 2002.
10. Loots, W. Smith, T.H.C. A parallel algorithm for the zero-one knapsack problem, International Journal Parallel Program, Vol. 21, pp. 313-348, 1992.
11. Martello, S., Toth, P. Knapsack problems: Algorithms and computer implementation, Willey and Sons, Chichester, 1990.
12. Martello, S., Toth, P. An upper bound for the zero-one knapsack problem and a branch and bound algorithm. European Journal of Operational Research, Vol. 1, pp. 169-175, 1977.
13. Penn, M., Hasson, D., Avriel, M. Solving the 0/1 proportional Knapsack problem by sampling, J. Optim. Theory Appl pp. 261-272, 1994.
14. Sahni, S. Approximate algorithms for the 0/1 knapsack problem, Journal of ACM, Vol. 22, pp. 115-124, 1975.
15. Vasquez, M., Hao, J.K. A hybrid approach for the 0/1 multidimensional knapsack problem, in Proceedings of the 13^{th} International Joint Conference on Artificial Intelligence pp. 328-333, 2001.

16. Zitzler, E., Thiele, L. Multiobjective optimization using evolutionary algorithms-a comparative case study. In Fifth International Conference on Parallel Problem Solving from Nature, A. E. Eiben, T. Back, M. Schoenauer and H. P. Schwefel Eds., Springer, Berlin, Germany, 1998, pp. 292-301.
17. Zitzler, E., Thiele, L. Multiobjective Evolutionary Algorithms: A comparative case study and the Strength Pareto Approach, IEEE Transaction on Evolutionary Computation, Vol 3, pp. 257-271, 1999.
18. Zitzler, E. Evolutionary algorithms for multiobjective optimization: Methods and Applications, Ph. D. thesis, Swiss Federal Institute of Technology (ETH) Zurich, Switzerland.
19. Zitzler, E., Laumanns, M., Thiele, L. SPEA 2: Improving the Strength Pareto Evolutionary Algorithm, TIK Report 103, Computer Engineering and Networks Laboratory (TIK), Department of Electrical Engineering Swiss federal Institute of Technology (ETH) Zurich, 2001.
20. http://www.tik.ee.ethz.ch/ zitzler/testdata.html

Part XIV

Control

Closed Loop Control for Common Rail Diesel Engines based on Rate of Heat Release

Nicola Cesario

[♯]SST Corporate R&D STMicroelectronics
via Remo de Feo, 1, Arzano(NA), 80022, Italy
nicola.cesario@st.com

Key words: Evolutionary algorithms, Evolution Strategies, Neural Networks, Clustering Analysis, Clustering Algorithms, Soft Computing Models for Automotive Applications, Features Extraction, Virtual Sensors.

1 Introduction

In the last years, the guideline for injection control systems of diesel engines is the realization of a micro-controller, which is able in real-time to find, through an optimization procedure oriented to decrease polluting gases emitted by the engines and the engine fuel consumptions, the “optimal” injection strategy related to driver load demand. The “optimal” injection strategy is associated with the optimal tradeoff among the following conflicting objectives: maximize torque, minimize fuel consumptions, reduce noise and polluting emissions (NOx and soot).

One of the most used injection control system is the map-based control. The main feature of this control systems is to link, at engine speed changing, driver demand (*loadDD*) and a set of parameters describing the injection strategy $(param_1, param_2, ..., param_n)$. From a mathematics viewpoint, this link can be formalized in the following way:

$$(param_1, ..., param_n) = f(speed, loadDD) \tag{1}$$

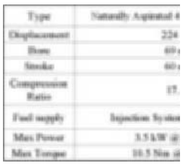
where the function domain is $\mathbb{R}^2_{[0,1]}$. This space has been transformed in a grid with M possible values of variable *speed* and P possible values of variable *loadDD*. In so doing, the function in 1 has been transformed into a set of n matrices (*control maps*). Each matrix chooses (according to driver load demand ($loadDD_p$) and according to the current value of engine speed ($speed_m$)) one parameter from amongst the n parameters of the injection strategy ($param_i$), see 2.

$$\tilde{f}^{(i)}_{m,p} = \tilde{f}^{(i)}(speed_m, loadDD_p) = param_i \tag{2}$$

where $i = 1, ..., n$, $m = 1, ..., M$ and $p = 1, ..., P$.

The following is the procedure for the set-up of a *control map*. At first, it is necessary to set map dimensions, that is, the number of matrix rows and columns $\tilde{f}^{(i)}_{m,p}$. Afterwards, for each load level and for each *speed* value you have to determine, according to experimental tests, the optimal injection strategy. We applied this heuristic procedure to a “test case”: injection control system for a common rail diesel engine supplied by a two injections strategy. For a description of engine properties see figure 1. In a typical map-based control system the real-time choosing of the appropriate injection strategy is obtained by a linear interpolation between parameter values ($param_1, ..., param_n$) included in the *control maps*.

A map-based injection control system is a *static* and *open loop* system. It is *static* because *control maps* are determined off-line by processing data acquired during experimental tests. Besides, an on-line updating of *control map* values is not expected. It is *open loop* because there is not a monitoring of the injection law obtained by the linear interpolation of *control map* values; in other words, for the current injection law, there is not a control of the emissions levels of NOx and soot.



Type	Naturally Aspirated 4
Displacement	224
Bore	69
Stroke	60
Compression Ratio	17.
Fuel supply	Injection System
Max Power	3.5 kW @
Max Torque	10.5 Nm @

Fig. 1. Features of our “test case” engine.

Fig. 2. *Static* and *open* injection control system.

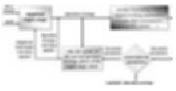


Fig. 3. *Dynamical* and *closed loop* injection control system.

Figure 2 describes the block scheme of a typical map-based injection control system. Figure 3 describes the block scheme of a typical map-based *dynamical* and *closed loop* system for the injection control. The later comprises the following blocks: engine *control maps*, models which forecast engine operating parameters, a set of thresholds for engine operating conditions and a set of rules (e.g. fuzzy rules) in order to update the current injection law and/or the values of the *control maps*. In this framework, it is normally added a “grey-box” model which is able to forecast **ROHR** versus injection strategy, engine speed and other quantities (such as temperature, air flow, etc...). From the time averaged **ROHR**, associated with a fixed operating condition of the engine, we can obtain qualitative and quantitative information on the combustion process, emissions, not to mention a model to rebuild the cylinder time averaged pressure cycle.

A feasible *dynamical* and *closed loop* control system, based on our “grey-box” model, could be realized in the following way. The controller reference signal could be the time averaged **ROHR**, estimated for the current operating condition of the engine by the "grey-box" model. The estimated **ROHR** could be compared with the real **ROHR** calculated by a simple derivative of the time averaged pressure cycle. In this case, the controller's aim would be to minimize this difference, modifying the fuel injection strategy control parameters selected by the *control maps*. In any case, the main requirement of this control system is to have a real cylinder pressure sensor.

2 Combustion Process in Diesel Engines: Rate of Heat Release

Modelling of the combustion process in internal combustion engines is based on the first principle of thermodynamics applied to the fluid included in the combustion chamber. In a simpler approach for modelling of the combustion process in internal combustion engines, fluid can be considered spatially uniform in composition, temperature and pressure. Neglecting fuel mass flows through the boundary surfaces of the chamber, heat flow released by the chemical reactions of the combustion ($\frac{dQ_b}{d\theta}$) is equal to:

$$\frac{dQ_b}{d\theta} = \frac{dE}{d\theta} + \frac{dL}{d\theta} + \frac{dQ_r}{d\theta} \tag{3}$$

where $\frac{dE}{d\theta}$ represents system internal energy variation, $\frac{dL}{d\theta}$ is the mechanical power exchanged with the environment by the piston moving and $\frac{dQ_r}{d\theta}$ is the heat consumed after the contact between the fluid and the cool sides of the chamber. Considering the fluid as a perfect gas with a mean temperature equal to T, we have that $E = mc_vT$; so doing and without mass flows, we have that:

$$\frac{dE}{d\theta} = mc_v\frac{dT}{d\theta} \tag{4}$$

The power transmitted to the piston is:

$$\frac{dL}{d\theta} = p\frac{dV}{d\theta} \tag{5}$$

Temperature is expressible, by the perfect gas equation, as a function of p and V:

$$T = \frac{pV}{mR} \tag{6}$$

deriving the last equation we have that:

$$\frac{dT}{d\theta} = \frac{p}{mR}\frac{dV}{d\theta} + \frac{V}{mR}\frac{dp}{d\theta} \tag{7}$$

Mixing the previous equations we obtain the following relation for the *Rate of Heat Release*:

$$\frac{dQ_b}{d\theta} = \Big[\frac{c_v}{R} + 1\Big] p\frac{dV}{d\theta} + \frac{c_v}{R} V\frac{dp}{d\theta} + \frac{dQ_r}{d\theta} \tag{8}$$

By a measurement of the cylinder pressure cycle, knowing the volume variation versus the crank angle $V(\theta)$ and using the equation 6, it is possible to determine fluid mean temperature trend. The last one is useful for modelling of heat losses through cylinder cool sides $\frac{dQ_r}{d\theta}$. Finally, replacing $V(\theta)$, $p(\theta)$ and $\frac{dQ_r}{d\theta}$ in 8 we can obtain rate of heat release as a function of the crank angle $(\frac{dQ_b}{d\theta})$. The integral between θ_i and θ_f, which are respectively the crank angle of the combustion start and combustion end, provides the heat quantity released in the combustion process. This is almost equal to the product of the fuel mass m_c and its "lower heating value" H_i.

$$Q_b = \int_{\theta_i}^{\theta_f} \Big(\frac{Q_b}{d\theta}\Big) d\theta \simeq m_c H_i \tag{9}$$

This approximation depends how the oxidation reactions evolve and the accuracy with which we analyzed the combustion process from an energetic viewpoint. Deriving by θ the logarithm of both sides of 9, we obtain the changing law of the fuel burnt mass $x_b(\theta)$.

$$\frac{1}{Q_b}\frac{dQ_b}{d\theta} = \frac{1}{m_c}\frac{dm_c}{d\theta} = \frac{dx_b}{d\theta} \Rightarrow \frac{dQ_b}{d\theta} = m_c H_i \frac{dx_b}{d\theta} \tag{10}$$

The fraction of fuel mass $x_b(\theta)$ has a S shape and it can be approximated by the following exponential function (*Wiebe function*):

$$x_b = 1 - exp\Big[-a\Big(\frac{\theta - \theta_i}{\theta_f - \theta_i}\Big)^{m+1}\Big] \tag{11}$$

with a suitable choice of the parameters a and m. The parameter a, known as the efficiency parameter, estimates the completeness of the combustion process. Instead m, known as the chamber shape factor, affects the combustion velocity. Typical values of a belong to the range $[4.605, 6.908]$; these values correspond to the completeness of the combustion process $(\theta = \theta_f)$ between the 99% and the 99.9% (that is $x_b \in [0.99, 0.999]$).

3 Effective ROHR Forecasting Model Requirements

Increasing the number of injections, the shape of **ROHR** becomes more complicated. There are many phases characterizing the combustion process and there are many factors affecting **ROHR** shapes. In this context, a model, which effectively forecasts the **ROHR**, must be *flexible* and *global*. It must be *flexible* in order to adapt itself to any fuel multiple injection strategy and to any shape of **ROHR**. Moreover, it must be *global*in order to simulate the time averaged **ROHR** associated to all engine operating conditions. In so doing, a similar model can be added in the *dynamical* and *closed loop* injection control scheme in figure 3.

4 Soft Computing Techniques Based Model Forecasting ROHR

Model design, that we propose to simulate the time averaged **ROHR**, basically comprises the following phases: *Choosing of the number of Wiebe functions on which we can break down **ROHR** signals*, *Ψ transform*, *Clustering on Ψ transform outputs*, *Evolutionary design of MLP neural network*, *Training and testing of MLP neural network*. In the first phase, we choose the Wiebe function set on which we can break down **ROHR** signals. In the second phase, with an analogous approach to [1], we find the optimal transform by which we can describe the signal of the time averaged **ROHR** by a restricted number of parameters (12).

$$\Psi(HRR(\theta)) = (c_1^k, c_2^k, \ldots, c_s^k) \qquad k = 1, 2, ...K \tag{12}$$

In the previous equation $HRR(\theta)$ indicates the time averaged **ROHR** signal acquired in the testing room for a fixed fuel multiple injection strategy and for a fixed engine operating point, while $(c_1^k, \ldots, c_s^k)$ $k = 1, ..., K$ are the K strings of s coefficients linked by the Ψ transform to the current signal. In the third phase, we determine using a clustering analysis the optimal coefficient strings; this analysis is performed according to the principles of the regularization theory of Tikhonov for "well-posed" problems (see [2]). The last phases of model design are dedicated to set-up, training and testing of an MLP neural network which has as inputs the system inputs $(speed, param_1, \ldots, param_n)$ and as outputs the associated strings selected in the previous steps. The last result is a "grey-box" model able effectively to rebuild the time averaged **ROHR** signal associated with a fixed fuel injection strategy and a fixed engine operating point. The neural network model determines the coefficients, which on the chosen function set (Wiebe functions set) describe **ROHR** signals.

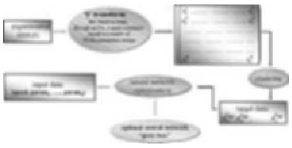


Fig. 4. The block scheme of the **ROHR** "grey-box" model.

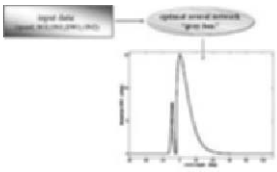


Fig. 5. Data flow of the **ROHR** "grey-box" model

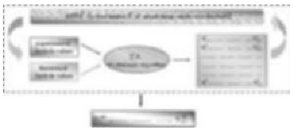


Fig. 6. Ψ transform block scheme.

Figures 4 and 5 describe the block scheme and data flow of the "grey-box" model.

4.1 Ψ Transform

Ψ transform (see the block scheme in figure 12) is based on evolutionary algorithms. It minimizes a fixed fitness function which is related to the fitting of the experimental **ROHR** on the chosen Wiebe function set. In this case, we used an $ES-(1+1)$ (Evolution Strategy-(1+1), see [3]) and as fitness function we used the mean square error which is related to the fitting of the experimental **ROHR** on the chosen overlap of Wiebe functions. These functions realize a function set suitable to break down **ROHR** signals.

For our test case in order to fit the time averaged **ROHR** signal, we used an overlap of two Wiebe functions. The former fits the **ROHR** "Pilot" while the latter fits the "Main". For this set of Wiebe functions, the number s of coefficients $(c_1^k, \ldots, c_s^k)$ is equal to 10; in other words, for each Wiebe function the evolutionary algorithm has to determine the following 5 parameters: a is the efficiency parameter of the combustion, m is the chamber shape factor, θ_i and θ_f are the crank angles describing respectively the combustion start and end, finally, m_c is the fuel mass (see 10 and 11). These parameters are related only to the part of the combustion process which is approximated by the current Wiebe function.

Increasing the number of Wiebe functions on which we analyze the experimental **ROHR**, the dimension of the parameter space, on which the evolutionary algorithm works, increases. The aim of the evolutionary algorithm is to find the optimal K strings of coefficients satisfying a fitting error threshold fixed by user. In this context, it is better to increase the initial population of strings of the evolution algorithm P and the minimum number of strings satisfying the threshold condition K. P indicates the number of coefficient strings randomly extracted in their definition range, K is the minimum number of population strings which must satisfy the threshold condition in order that the algorithm ends its execution. If the algorithm converges without that K strings attain the threshold condition, it executes itself another time with an increased P. Process ends when K strings satisfy the threshold condition fixed by user. Several tests suggest for the variables P, K and $\varDelta P$ the following values: $P = 50W_n$, $K \in [5W_n, 10W_n]$ and $\varDelta P = 0.1P$. In the previous relationships W_n is the number of Wiebe functions chosen to break down the **ROHR** signal. An evolution algorithm, such as $ES-(1+1)$, converges when all P strings (see 6), for a fixed number of iterations t_{min}, do not improve their fitness, that is

$$\left|\frac{\varDelta f_j^{t,t+1}}{f_j^t}\right| \leq Er_{conv} \quad j = 1, 2, ...P \tag{13}$$

In 13 $\varDelta f_j^{t,t+1}$ describes the $j-$th individual fitness variation going by t step to $t+1$ step of the algorithm, Er_{conv} represents the $j-$th individual maximum incremental fitness variation in order that the algorithm converges. The equation 12 suggest that the output of $\varPsi$ transform is not univocal. Indeed, after fixing a threshold for the fitting error of the experimental **ROHR**, there are exactly K strings of coefficients $(c_1^k, \ldots, c_s^k)$ whereby we have a fitting of the experimental **ROHR** with an error minor or equal to the threshold.

4.2 Clustering

In the second phase of model design, the coefficient matrix $(c_1^k, \ldots, c_s^k)$, associated by Ψ transform to the input data $(speed, param_1, \ldots, param_n)$, are processed by a clustering algorithm (see figure 4).

The aim is to identify the "optimal" strings of coefficients $(c_1^{k_{opt}}, \ldots, c_s^{k_{opt}})$ which involve similar variations between input and output data (output data are the strings of coefficients). Our "grey-box" model is an MLP neural network. This neural network is trained on a database containing the experimental inputs and the associated target data $(c_1^{k_{opt}}, \ldots, c_s^{k_{opt}})$. We performed the neural network training avoiding that our model goes in over-fitting; in other words, we trained neural network maximizing model generalization capability. The last result is a "grey-box" model which effectively forecasts the coefficient string $(c_1^k, \ldots, c_s^k)$ related to a fixed input. These strings of coefficients, on the chosen set of Wiebe functions, allow us to rebuild the time averaged **ROHR** signal (see 5).

For a more detailed comprehension of the topics described before, it is necessary to describe design process of a neural network from a mathematical viewpoint. A neural network with a scalar output can be described as an hypersurface Γ in $\mathbb{R}^{m+1}$ space, that is, a map $s : \mathbb{R}^m \to \mathbb{R}^1$, see [4]. In this formalism, the index m represents the design space dimension. Neural network design can be described as a multivariable interpolation in a high dimensional space. Given a set of N different points $\{\mathbf{x}^{(i)} \in \mathbb{R}^m | i = 1, 2, ..., N\}$ and a corresponding set of N real numbers $\{d_i \in \mathbb{R}^1 | i = 1, 2, ..., N\}$, it is necessary to find a function $F : \mathbb{R}^m \to \mathbb{R}^1$ that satisfies the interpolation conditions (this problem is known as the problem of reconstructing the mapping F):

$$F(\mathbf{x}^{(i)}) = d_i, \quad i = 1, 2, ..., N \tag{14}$$

In our framework, the relationships of 14 change because neural network has vectorial outputs. Generally, the above mentioned problem of model design is an "ill-posed" problem. In fact, the unavoidable presence of noise or imprecision in real-life training data adds uncertainty to the reconstructed input-output mapping and increases the probability that one of the three conditions characterizing a "well-posed" problem is violated. For this purpose, we list three conditions to be satisfied in order that the problem of reconstructing the mapping $f(X) \to Y$ is "well-posed" (see [2] and/or [4]):

- *Existence*, $\forall x \in X \; \exists y = f(x)$ where $y \in Y$
- *Uniqueness*, $\forall x, t \in X$ we have that $f(t) = f(x) \Leftrightarrow x = t$
- *Continuity*, $\forall \epsilon > 0 \; \exists \delta = \delta(\epsilon)$ such as $\rho_x(x, t) < \delta \Rightarrow \rho_y(f(x), f(t)) < \epsilon$

In the previous relationships, the symbol $\rho_x(.,.)$ represents the distance between its arguments in the reference vectorial space. If only one among three conditions is violated then the problem of model design is not "well-posed". This means that only a restricted part of training data set is used effectively for the model set-up. Nevertheless, there is a consolidated theory called the regularization theory for solving "ill-posed" problems (see [2]). The basic idea of the regularization theory is to stabilize the solution (i.e. the map $f(X) \rightarrow Y$) so that Δx has a size similar to Δy. This means that the solution to the reconstruction problem must be *smooth*, in other words similar inputs must correspond to similar outputs. All that can be done choosing strings of coefficients $(c_1^{k_{opt}}, \ldots, c_s^{k_{opt}})$ so that:

$$\sum_{i,j=1}^{N_{tot}} |\Delta x_{ij} - \Delta y_{ij}^{opt}| = min(\sum_{k,h=1}^{K} \sum_{i,j=1}^{N_{tot}} |\Delta x_{ij} - \Delta y_{ij}^{k,h}|) \qquad (15)$$

where $\Delta x_{ij} = |(speed^{(i)}, param_1^{(i)}, \ldots, param_n^{(i)}) - (speed^{(j)}, param_1^{(j)}, \ldots, param_n^{(j)})|$ and $\Delta y_{ij}^{k,h} = |(c_1^{k,(i)}, \ldots, c_s^{k,(i)}) - (c_1^{h,(j)}, \ldots, c_s^{h,(j)})|$. Given a set of input data, $(speed^{(i)}, param_1^{(i)}, \ldots, param_n^{(i)}) i = 1, \ldots, N_{tot}$, there are $K^{N_{tot}}$ possible strings of coefficients which could be related by the Ψ transform to the input data. Hence, from a computational viewpoint, the lowest expensive way to find the minimum of the sum in the relation of 15 is to use evolution algorithms. The $j-$th individual of the evolution algorithm population is a set of N_{tot} strings, each of them containing s coefficients. This set is selected from amongst the possible $K^{N_{tot}}$. From a viewpoint of the clustering analysis, the choosing of optimal strings $(c_1^{k_{opt}}, \ldots, c_s^{k_{opt}})$ is like the extraction of barycenters from the distribution of N_{tot} clusters.

4.3 Set-up of the "Grey-Box" Model

The last phase of the "grey-box" model set-up is the training of an MLP neural network on a set of N_{tot} input-output couples. Target data for the supervised learning are strings of coefficients $(c_1^{k_{opt}}, \ldots, c_s^{k_{opt}})$ selected in the previous phase of clustering. We did not design the MLP neural network in an heuristic way. Both the number of neurons of the hidden layer and the regularization factor of the performance function (see [4]), are model endogenous parameters. The research space of these parameters is $\mathbb{R}_{[0,1]}^{N1}$, where the symbol $\mathbb{R}_{[0,1]}$ represents real numbers in the compact range $[0, 1]$ and $N1$ is the total number of model endogenous parameters. An exhaustive research in a similar space is not possible, this is because for this research we used some EA, such as ES-$(1 + 1)$ (Evolution Strategy). A trivial choice of the fitness function could be the estimate error on testing data set of our "grey-box" model. In so doing, we could not have a "generalized" estimate of the model endogenous parameters. "Generalized" estimate means that the choice of the model endogenous parameters is made to increase the model generalization capability, that is, the model "generalized forecast capability"; for a more detailed description of the "generalized forecast capability" of a model see [5].

Really, the shape of the fitness function, which we used, is the following:

$$V_0(S_1, S_2, ..., S_{N1}) = \frac{1}{N^*}\sum_{i=1}^{N^*}\frac{1}{N}\sum_{k_i=1}^{N}[\lambda_{k_i} - F^i_{(S_1,S_2,....,S_{N1})}(\mathbf{x}^{(k_i)})] \quad (16)$$

where $N^* = \begin{pmatrix} N+N1 \\ N \end{pmatrix}$. The formula in 16 derives from an our generalization of the *ordinary cross-validation estimate* of endogenous parameters of a neural network, see chapter 5 of [4]. The parameter N^* is the number of possible choices of a testing set with N samples in a data set composed by $N+N1$ input-output couples. In 16, k_i labels the N elements of the testing data set selected by the i-th choice. Model output is described by the symbol $F^i_{(S_1,S_2,....,S_{N1})}$. In 16 the symbol S_i indicates the $i-$th endogenous parameter of our neural network model. The optimal choice of the model endogenous parameters is the one which minimizes the functional $V_0(S_1, S_2, ..., S_{N1})$. To search this minimum we used the before mentioned evolutionary algorithms.

The last result is a "grey-box" model able to forecast, from a fixed fuel multiple injection strategy and a fixed engine operating point, the string of coefficients which, on the chosen set of Wiebe functions, rebuilds the time averaged **ROHR** signal (5).

5 Test Case

We applied our "grey-box" model to the following test case: a common rail diesel engine supplied with a double injection of fuel. Figures 7, 8 and 9 illustrate preliminary outcomes of our work, that is the reconstruction of the time averaged **ROHR** of our test case engine for some fixed engine operating conditions.

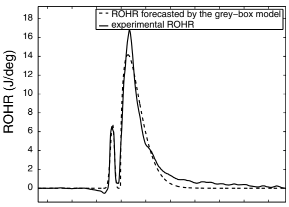


Fig. 7. ROHR simulated by the "grey-box" model.

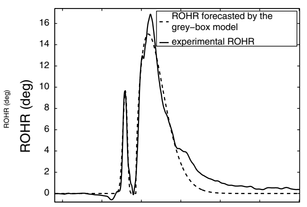


Fig. 8. ROHR simulated by the "grey-box" model.

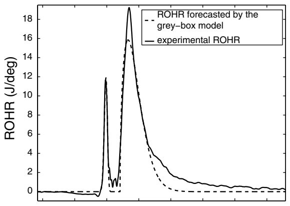


Fig. 9. ROHR simulated by the "grey-box" model.

Figure 10 shows the reconstruction of the time averaged pressure cycle which is related to a fixed **ROHR**. This proves that our "grey-box" model has a satisfactory forecast capability.

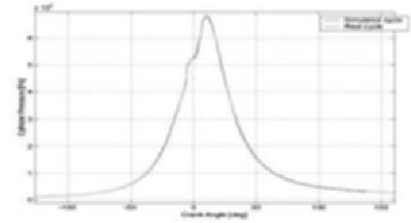


Fig. 10. Time averaged cylinder pressure cycle simulated by the **ROHR** "grey-box" model.

6 Conclusions

The "grey-box" model, that we proposed to simulate the time averaged **ROHR** in common rail diesel engines, is a *global*, *portable* e *multi-resolution* "single zone" model. *Global* means the model is able to forecast the time averaged **ROHR** for any engine operating condition. *Portable* means it is applicable to any diesel engine. *Multi-resolution* means the user can improve the model accuracy by increasing the number of Wiebe functions used to analyze the **ROHR** signal. This last feature does not affect, in a meaningful way, the computational weight of the model set-up.

The "grey-box" model can be used to realize a *dynamical* and *closed loop* injection control system for common rail diesel engines. A similar control system could be able to detect *off-design* operating conditions of the injection system and engine; for example, it could be detect whether one cylinder works in bad conditions, this could be related to an improper functioning of the injection system. Moreover, this control system can update control maps during the engine life. In the control system scheme, the controller reference signal is the **ROHR** estimated by our model. The estimated **ROHR** is compared with the real **ROHR** calculated on the current cylinder pressure cycle. The controller aim is to minimize this difference modifying the control parameters.

References

1. S. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Pattern Anal. and Machine Intell.*, 11(7):674–693, 1989.
2. A.N. Tikhonov and V.Y. Arsenin. *Solution of Ill-posed Problems.* W.H. Winston, Washington, DC, 1977.
3. H. G. Beyer. *The Theory of Evolution Strategies.* Springer-Verlag, New York, 2001.
4. S. Haykin. *Neural Networks: A comprehensive Foundation.* Prentice Hall International Editions, Piscataway, 1999.
5. V.N. Vapnik. An overview of statistical learning theory. *IEEE Trans. On Neural Networks*, 9.

A MIMO Fuzzy Logic Autotuning PID Controller: Method and Application

Otacílio M. Almeida, Laurinda Lucia N. Reis; Luis Daniel S. Bezerra, Sanderson Emanuel U. Lima

Dept. of Electrical Engineering, Federal University of Ceara
Fortaleza - Ceara, Brasil

Abstract. — In this paper a new autotuning fuzzy *PID* control method for *SISO* and *MIMO* systems is proposed. The fuzzy autotune procedure adjusts on-line the parameters of a conventional PID controller located in the forward loop of the process. Fuzzy rules, employed to determine the set of PID gains, are based on the representation of human expertise on how can be the behaviour of gain and phase margins of a control system to efficiently compensating the system errors. The proposed control scheme offers advantages over the conventional fuzzy controller such as: *i*) a systematic design is attained in both SISO and MIMO cases; *ii*) it is necessary only one rule base for all loops; *iii*) the tuning mechanism is simple and control operators can easily understand how it works; and *iv*) it is completely autotuned, requiring only one relay feedback experiment per loop. Simulation examples and a practical essay are assessing the effectiveness of the proposed control algorithms.

Keywords: fuzzy control, PID control, multivariable feedback control, nonlinear systems.

1 Introduction

For years the fuzzy logic control has proved its broad potential in industrial applications (Lee, 1990; Bonissone, 1994; Altrock and Gebhardt, 1996; Qin, *et al*., 1998). The fuzzy control theory has been applied to a number of systems with single-input and single-output (*SISO*) structures, mainly to overcome uncertain parameters and unknown models (Hu *et al*, 1999; Yager and Filev 1994;). Generally, fuzzy control shows good performance for controlling nonlinear and uncertain systems that could not be controlled satisfactorily by using conventional controller, for example, a conventional *PID* controller (Ying, *et al*., 1990). Also, in applications when there are multiple-input and multiple-output (*MIMO*) systems with strong loop interactions, conventional controllers do not work well and advanced control conceptions are required. Control literature of *MIMO* fuzzy logic controllers (*MIMO FLC*) shows limited results and a great effort has been used by researchers to derive stable control strategies. Usually, a *MIMO FLC* is tuned by trial-and-error that means a tedious and time-consuming task, and design techniques for systematic tuning must be obtained. Also, *MIMO* FLC applications are frequently solved by using the conventional decoupling theory and with single FLCs, resulting in high-dimensional rule-bases that may not be implemented in practical systems, due to required processing time (Nie, 1997).

In this paper a method for autotuning *SISO* and *MIMO FLC* is proposed. The autotune procedure adjusts on-line the parameters of a conventional *PID* controller located in the forward loop of the process.

In order to give an autotuning capacity to *SISO* and *MIMO* cases, a scheme of multivariable identification is implemented by using relay feedback (Wang and Shao, 1999; Luyben, 1987; Shen and Yu, 1994; Shiu and Hwang, 1998). A *SISO* second-order plus dead-time transfer function for loop is adopted as plant model. Also, fuzzy rules, employed to determine the set of *PID* gains, are based on the representation of human expertise on how must be the behavior of gain and phase margins of a control system to efficiently compensating the system errors. In both *SISO* and *MIMO* cases, gain and phase margins are determined by a set of Mandani rules and the membership function of the fuzzy sets are based on the system error and its difference.

Performance and robust stability aspects are assessed by practical and simulated examples of *SISO* and *MIMO* systems. Simulation results for other conventional control algorithms are also included for comparison purpose. The proposed control scheme offers advantages over the conventional fuzzy controller such as: *i*) a systematic design is attained in both *SISO* and *MIMO* cases; *ii*) it is necessary only one rule base for all loops; *iii*) the tuning mechanism is simple and control operators can easily understand how it works; and *iv*) it is completely autotuned, requiring only one relay feedback experiment per loop.

In this paper a method for autotuning *SISO* and *MIMO FLC* is proposed. The autotune procedure adjusts on-line the parameters of a conventional *PID* controller located in the forward loop of the process.

The outline of this paper is as follows. Fuzzy *PID* structure for the *SISO* case is described in section 2, where simulated, experimental and comparative examples are given to assess the design performance. *MIMO* case is shown in section 3. A brief conclusion is given in section 4.

2 FUZZY *PID* CONTROLLER (*FPID*) *SISO* CASE

The *FPID* controller, proposed by Almeida and Coelho (2001, 2002), uses relationships among the gain margin (*Am*), the phase margin (*Φm*), the *PID* parameters (Kc, Ki, Kd), and a second-order process model. Fuzzy rules are based on the representation of human expertise on how can be the behavior of gain and phase margins of a control system to efficiently tracking setpoint and pertubance rejection.

2.1 FPID SISO Structure

In *FPID* design the gain margin A_m and the phase margin Φ_m, given by Eqs. (1) and (2), are considered linguistic variables which values are defined with respect to the same universe of discourse specified by human expertise about the operational knowledge of the process.

$$\Phi_m = \frac{\pi}{2}\left(1 - \frac{1}{A_m}\right) \qquad 1$$

and a second-order process model parameters (*a*, *b*, *c*) as given by the following equations:

$$\begin{bmatrix} K_c \\ K_i \\ K_d \end{bmatrix} = \frac{\pi}{2A_m L}\begin{bmatrix} b \\ c \\ a \end{bmatrix} \qquad 2$$

Also, it is assumed that the feedback system gain and phase margins are in prescribed ranges [$A_{m,min}$, , $A_{m,max}$] and [$\Phi_{m,min}$, $\Phi_{m,max}$], respectively. For convenience, values of A_m are normalized into a range between zero to one by the following linear transformation, Eqs. 3 and 4:

$$A_m^{'} = (A_m - A_{m,\min}) / (A_{m,\max} - A_{m,\min}) \qquad 3$$

$$\Phi_m^{'} = (\Phi_m - \Phi_{m,\min}) / (\Phi_{m,\max} - \Phi_{m,\min}) \qquad 4$$

where $A_m^{'}$ and $\Phi_m^{'}$ are normalized gain and phase margins, respectively. Values of $A_m^{'}$ are determined by a set of fuzzy rules of the form

R_i: If *e(t)* is *A* and *Δe(t)* is *B* then $A_m^{'}$ is *C* 5

where A_m is the gain margin for R_i rule(*i*=1 to n), *A*, *B* and *C* are fuzzy sets on the corresponding supporting sets.

Membership functions of these fuzzy sets for *e(t)* and *Δe(t)* are shown in Fig.1c. Fuzzy rule base sets are obtained from operator's expertise by using the step response of the process. Figs. 1a and 1b show an example of a desired time response and a fuzzy rule base.

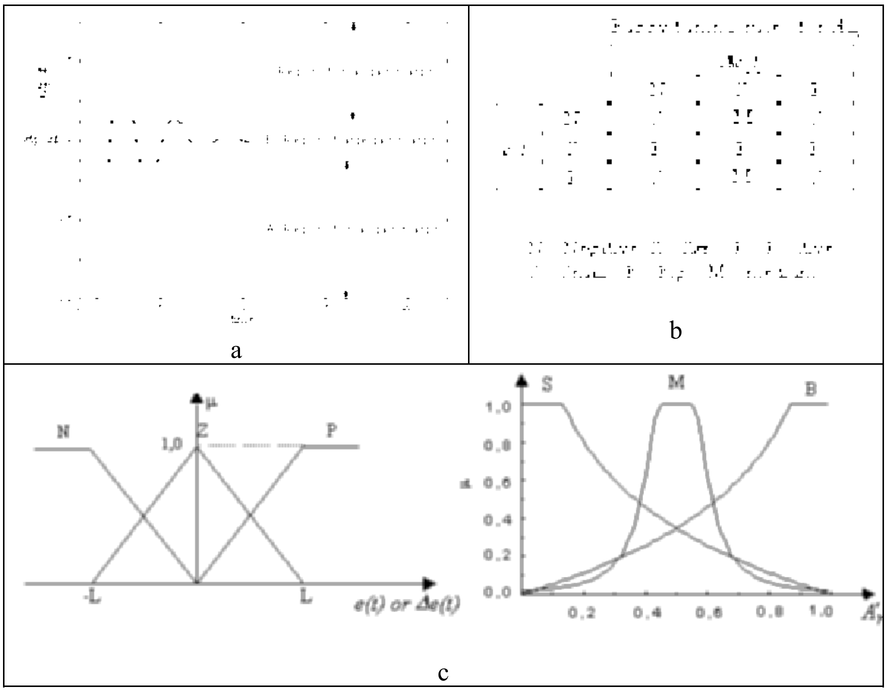


Figure 1. (a) Plant response and fuzzy rule base sets; (b) rule base; (c) membership functions.

The fuzzy set may be either Big, Small, Medium and it is characterized by logarithmic membership functions. The grade of the membership μ and the variable $A_m^{'}$ has the following relation:

$$\mu_B(A_m^{'}) = -\tfrac{1}{\eta}\ln(1-A_m^{'}) \qquad 6$$

$$\mu_M(A_m^{'}) = 1 - e^{-(\delta / |0.5-A_m^{'}|)^{2.5}} \qquad 7$$

$$\mu_S(A_m^{'}) = -\tfrac{1}{\eta}\ln(A_m^{'}) \qquad 8$$

where η and δ are adjustable parameters. In Fig. 2c η=4 and δ=0.1.

The truth value of the i^th^ rule in Eq. (5) μ_i is obtained by the product of the membership function values in the antecedent part of the rule (Nie, 1997).

$$\mu_i = \mu_{Ai}[J(t)] \cdot \mu_{Bi}[\Delta J(t)] \qquad 9$$

Based on μ_i, values of $A_m^{'}$ for each rule are determined from their correspondent membership function. The implication procedure is shown in Fig. 2.

Fig. 2. Implication procedure of a fuzzy rule.

By using the membership functions in Figs. 2 an 3, the following condition holds:

$$\sum_{i=1}^{n} \mu_i = 1 \tag{10}$$

Then, the defuzzification process, Fig. 2, yields

$$A'_m = \sum_{i=1}^{n} \mu_i A'_{m,i} \tag{11}$$

Once A'_m is obtained, A_m is calculated from Eq. (11) and *PID* parameters are derived from Eq. (2).

2.2 Simulation and Experimental Results for the FPID-SISO

The new autotuning fuzzy *PID* controller in SISO case is now assessed for its ability to control a nonlinear process given by Eq. (12). The relay feedback method is used to tune a modify Ziegler_Nichols PID parameters around an operational point. Output, control and reference signals of the process when controlled by PID and FPID controllers are shown in Fig. 3.

$$\dot{y}(t) = -y(t) + sin^2(\sqrt{|y(t)|}) + u(t) \tag{12}$$

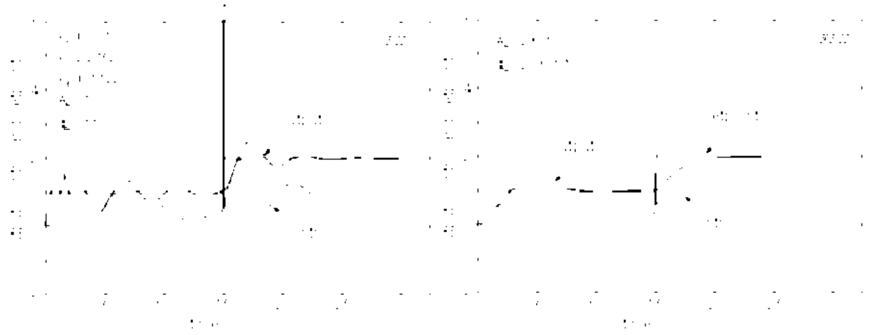


Fig. 3. Responses for (a) PID (left) ; (b) FPID (right).

It is evident that the *PID* presents poor performance while a well damped setpoint response is achieved by *FPID* controller.

In order to compare the servo performance and perturbation rejection capability of the *FPID* with *PID* the overshoot (M_p), damping index (M_ζ) and settling time (T_s) are measured according to Fig. 5.

Fig 4. Parameters for performance measurement: (a) servo behavior ; (b) disturbance rejection.

Table 1. Indices for *FPID* and *PID* .

	Time from 0 s to 15 s		
	M_p	M_ζ	T_s (s)
FPID	31.05%	82.15%	7.8
PID	75.57%	71.12%	>15
	Time from 15 s to 30 s		
	M_p	M_ζ	T_s (s)
FPID	2.53%	10%	4.9
PID	48.54	50.55	9.7

According to Fig. 3 and Table 1, before the model rupture the *FPID* performs better than *PID*.

In order to conclude monovariable performance tests, *PID* and *FPID* control approaches are assessed in only one single column of a double tank process. The coupled double tank plant was implemented by Department of Electrical Engineering of Federal University of Ceara and details of the process are available in http://www.gpar.dee.ufc.br/. It is a two-input two-output system. Each tank has an inlet commanded through a voltage controllable pump (controlled variable) and an outlet that can be adjusted through a manually controlled valve. The outlets communicate to a reservoir from which the pumps extract the water to deliver into the tanks. The tanks are connected through a baffle valve which can adjusted manually. The pumps control is a PWM signal with duty cycle from 0 to 100% and frequency of 8KHz. The water level of tanks (controlled variable) are measured by a sensor transducer system with the real minimum achieved about 2cm and the maximum is about 1m. This makes the effective measurement range 0,1V to 5V. The analog transducers signal are read through a data acquisition card connected to the parallel PC Port and controlled by the MATLAB/Simulink software. Because of the physics of the system, the actual system dynamics are mildly non-linear and dead time. Furthermore the physical constraints mean that essentially inputs and outputs are all subject to saturation.

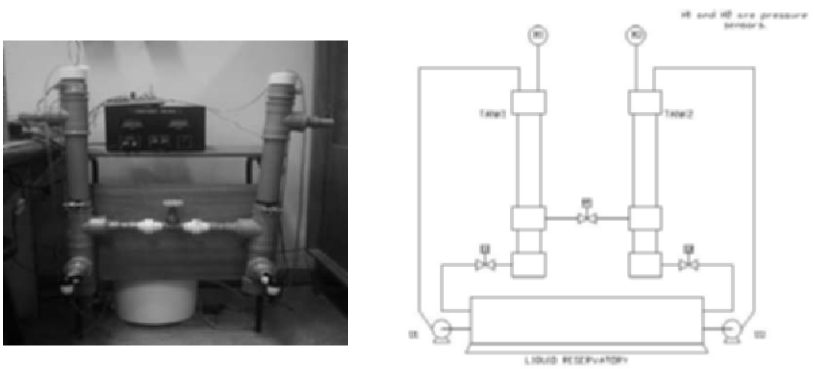


Figure 5. Experimental double tanks plant.

Step responses of *FPID–SISO* and *PID* controllers for setpoint canges are given in Fig. 6.

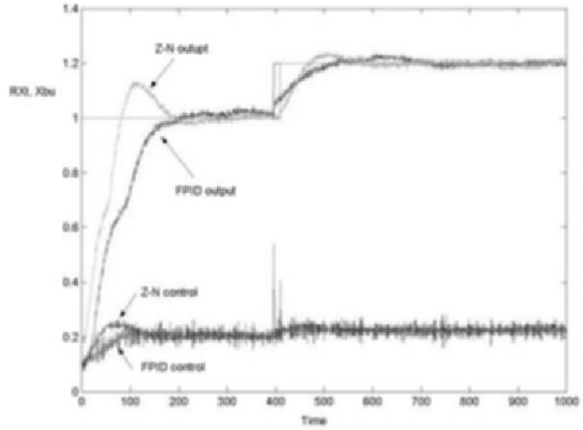


Figure 6. Tunnel heating responses for *FPID-P* and *GPC-PID*; load disturbance.

According to Fig. 6, both *FPID* and *PID* controllers give good control performance. Theses results show that the *FPID-SISO* is better than ZN improved method. For setpoint responses, the overshoot, the level of interactions and the settling time are improved with more stable responses.

3 FUZZY *PID* CONTROLLER: *MIMO* CASE

The multivariable fuzzy logic controller proposed in this paper utilizes fuzzy rules to determine the set of *PID* parameters. As in the *SISO* case, control signals in the *MIMO* case are generated by *PID* controllers,

3.1 A. FPID MIMO Structure

With additional conditions, the proposed *FPID SISO* controller can be generalized for the decentralized *MIMO FPID* case in a straightforward way. Fig. 6 shows the block diagram.

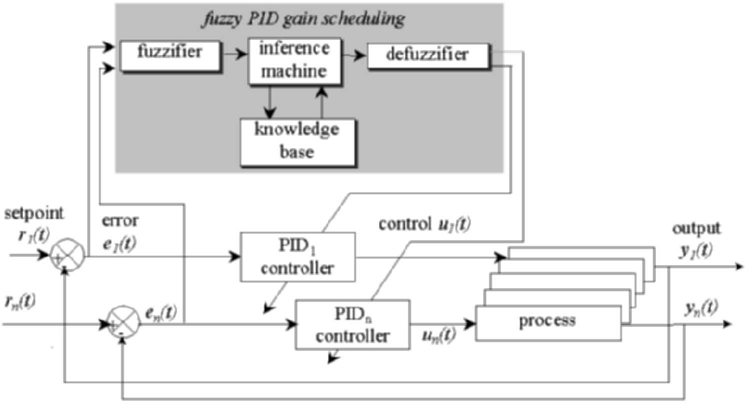


Fig. 6. Fuzzy gain scheduling PID controller: MIMO case.

MIMO fuzzy rules are the representation of human expertise on how must be the behavior of the gain and phase margins of a MIMO control system to efficiently compensating the system errors.

If $e_i(k)$ is A_v and $\Delta e_i(k)$ is B_v then $A_{m,j}$ is C_v ; v=1...p and i=1...n 13

where $A_{m,i}$, is gain margin for i loop, A_v, B_v and C_v are fuzzy sets on the corresponding supporting sets, p is the number of fuzzy sets and n is the number of loops in the multivariable system. The membership function (MF) of the fuzzy sets $e_i(k)$ and $de_i(k)$ as well the rule base are generalization of the *SISO* case. *MIMO PID* parameters are determined to ensure adequate gain and phase margins to the system.

Considering that sequential design is addressed, the *MIMO PID* parameters are tuned as in a *SISO* case. Considering a two-input/two-output (*TITO*) system, as example of a *MIMO* transfer function, *PID MIMO* controller can be written as

$$G_{c,i}(s) = k_i\left(\frac{A_i s^2 + B_i s + C_i}{s}\right) \qquad i = 1, 2 \tag{14}$$

where A_i, B_i, C_i and k_i are determined by the same *SISO* case strategy. MIMO PID parameters for a TITO system is given by

$$\begin{bmatrix} K_{p,i} \\ K_{i,i} \\ K_{d,i} \end{bmatrix} = \frac{\pi}{2LA_{m,i}} \begin{bmatrix} b_i \\ c_i \\ a_i \end{bmatrix} \qquad i = 1, 2 \tag{15}$$

where the parameters $A_{m,i}$ is a nonlinear variable determined by the fuzzy engine.

Design steps for the auto-tuning *MIMO* fuzzy controller are *i*) tune a Ziegler-Nichols *PID* controller considering individuals loops; *ii*) if the control system is unstable, one step of a sequential design should be done to tune a *PID* controller for the *MIMO* process; *iii*) identify the *MIMO* system by performing a relay experiment in each loop while the other loops are under PID controllers designed in step one. A transfer function like that in Eq. (18) is obtained for each loop; *iv*) define the discussion universe to the fuzzy variables; *v*) specify the maximum and minimum values to A'_m. Typical values for A'_m ranges from 2 to 5 and is corresponding to phase margin between 30 to 45; *vi*) apply the *MIMO* fuzzy controller engine.

Each control loop of the *FPID MIMO* controller runs with a independent *FPID* per loop. Since only one rule base is used to control whole loops, the reduced dimension of the rule base as well its simplicity correspond to the most important features of the proposed fuzzy scheme.

As in the *SISO* case, in the *MIMO* design, despite the loops interactions, a second order model is employed for each loop. The model of each loop can be determined from sequential relay experiment as in Almeida and Coelho (2002) or Shen and Yu, (1994). Initial PID pre-tune parameters can be determined by Ziegler-Nichols method.

3.2 Results for the *FPID- MIMO* – Double Tanks Case

In this section the double tank plant is used to evaluate the *FPID-MIMO* conception on a real time *MIMO* process. Manipulated variables are the input voltage to the pump motors, $U(s)=[u_1(s) \;\; u_2(s)]'$, and the water levels, $H(s)=[h_1(s) \;\; h_2(s)]$, are the controlled variables. The double tank transfer function $G(s)$ is not available, except by using a identification methods. However, by simulation accesses the column presents strong loop interactions.

According to step one of the proposed algorithm, Ziegler-Nichols *PID* (ZN-*PID*) (Luyben, 1990) parameters must be used as a pre-tuning and are given in Table 2. Parameters of the identified transfer function under relay experiments are shown in Table 2.

Table 2 – ZN pretune parameters.

	Parameters		
Loops	K_p	T_i	T_d
1	0.1543	10.2299	2.5575
2	0.0606	7.3630	0.2630

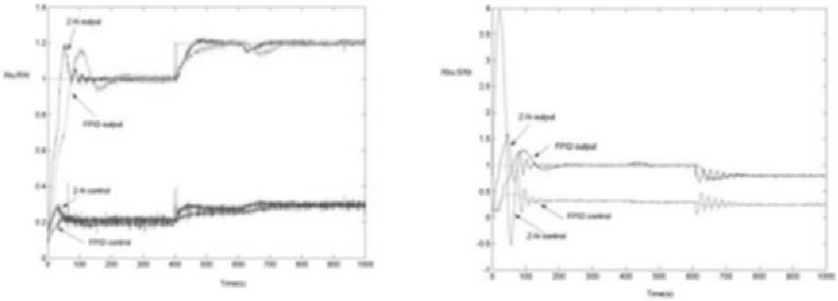


Figure 7. Control, output and setpoint for *WB* column (a) pair U_t - H_1 (b) pair U_2-H_2.

According to Fig. 7, both *FPID MIMO* and *PID MIMO* controllers performs well for double tank process control. However the *FPIP* controller gives better disturbance rejection on loop U_2-H_2 due to setpoint change on loop U_t - H_1.

Fig. 7 compare output, setpoint and control signals for double tank process under ZN improved control (Luyben, 1990) and *FPID-MIMO* control methods. Simulation results show that the *FPID-MIMO* is better than ZN method. For setpoint responses, the overshoot, the level of interactions and the settling time are improved with more stable responses.

4 Conclusion

A systematic method has been developed to design a fuzzy *PID* controller for *SISO* and *MIMO* cases. The method is based on gain and phase margin specifications and needs system identification under relay experiments. In the SISO case the gain is considered a fuzzy variable and in the *MIMO* case the sequential design is addressed in order to be possible the fuzziness of the gain and phase margins. The fuzzy *PID* controller derived successfully demonstrated better performance than the conventional *PID* controller for many case studies, particularly for nonlinear plants. The fuzzy *PID* controller is also able to tolerate many poor selections or inadequate implementation of the controller gains, for example, bad tuning for initial parameters of *PID* controller, which would make most conventional controller unstable.

The main goal of this paper is to provide to the plant operators with easy-to-understand fuzzy *PID* method for quickly achieving satisfactory control over unknown monovariable and multivariable systems. Despite its simplicity, the proposed method yielded monovariable and multivariable designs and a superior behavior to that resulting from empirical method based on trial-and-error procedure. The method has shown be adequate for practical applications, both in *SISO* and *MIMO* cases.

References

Almeida, O. M., Coelho, A. A. R.; (2001). Controlador *PID* com Escalonamento Nebuloso dos Ganhos: Auto-Sintonia, Análise e Implementação*, V Simpósio Brasileiro de Automação Inteligente – V SBAI.*

Almeida, O. M., Coelho, A. A. R.; (2002). A Fuzzy Logic Method for Autotuning a *PID* Controller: *SISO* and *MIMO* Systems, *15th IFAC World Congress on Automatic Control (*to appear*).*

Altrock, C and Gebhardt, J. (1996). Recent successful fuzzy logic applications in industrial automation*, Proc. of Fifth IEEE International Conference on Fuzzy Systems*, vol. 3, pp. 1845-1851.

Bonissone, P. P. (1994). Fuzzy logic controllers: an industrial reality, *Computational Intelligence: Imitating Life*, Zurada, J.M.; Marks II, R.J.; Robinson, C.J. (eds.), Piscataway, NJ, IEEE Press, pp. 316-327.

Hu, B.; Mann, G. K. I. and Gosine, R. G. (1999). New methodology for analytical and optimal design of fuzzy PID controllers, *IEEE Transaction on Fuzzy Systems* vol. 7, pp. 521-539.

Kim, E.; Lee, H. and Park, M. (2000) Limit-cycle prediction of a fuzzy control system based on describing function method, *IEEE Transaction on Fuzzy Systems,* vol. 8, pp. 11-22.

Qin, Y.; Jia, L. -M. and Zhang, X. -D. (1998). Automation of combustion process for coke oven using multivariable fuzzy control technique, *IEEE World Congress on Computational Intelligence*, Proc. of FUZZY-IEEE, Anchorage, Alaska, USA, pp. 606-610.

Luyben, W. L. (1990). *Process modeling simulation, and control for chemical engineers*, McGraw-Hill, Inc.

Nie, J. (1997). Fuzzy control of multivariable nonlinear servomechanisms with explicit decoupling scheme, *IEEE Transactions on Fuzzy Systems*, vol. 5, pp. 304-311.

Shen, S. –H. and Yu, C. –C. (1994). Use of relay-feedback test for automatic tuning of multivariable systems, *AIChE Journal*, vol. 40, pp. 627-646.

Shiu, S.-J. and Hwang, S., -H. (1998). Sequential design method for multivariable decoupling and multiloop PID controllers, *Ind. Eng. Chem. Process Des. Dev.*, vol. 37, pp. 107-119.

Ying, H.; Siler, W. and Buckley, J. J. (1990). Fuzzy control theory : a nonlinear case, *Automatica*, vol. 23, pp. 513-520.

Wang, Y. G. and Shao, H. H. (1999). PID tuning for improving performance, *IEEE Transactions on Control Systems Technology*, vol. 7, pp. 457-465.

Yager, R. R. and Filev, D. P. (1994). *Essentials of fuzzy modeling and control,* John Wiley & Sons, Inc., New York, N.Y.

Wood, R. K. and Berry, M. W. (1973). Terminal composition control of a binary destillation column, *Chem. Eng. Sci*., vol. 28, pp. 1707-1717.

Performance of a Four Phase Switched Reluctance Motor Speed Control Based On an Adaptive Fuzzy System: Experimental Tests, Analysis and Conclusions

Silviano Rafael[1], A.J. Pires[1], P.J. Costa Branco[2]

[1]LabSEI, Escola Superior de Tecnologia de Setúbal / Instituto Politécnico de Setúbal, Rua Vale de Chaves, Estefanilha, 2914 Setúbal, Portugal, srafael@est.ips.pt, apires@est.ips.pt, Tel: +351 265 790 000

[2]Instituto Superior Técnico /Universidade Técnica de Lisboa, Av. Rovisco Pais, 1096 Lisboa, Portugal, pbranco@alfa.ist.utl.pt , Tel: +351 218 417 432

Abstract. In this paper, the controller's tuning and performance of a speed controller prototype for switched reluctance motor is presented using significant experimental tests. The system uses an on-line learning mechanism to acquire and modify, if needed, the "good" fuzzy control rules. Experimental essays are analyzed and discussed in order to reveal some advantages of having a learning speed controller for the SR machine, and also the drawbacks that the use of using these controllers can introduce to the drive system and possible ways to overcome them.

Keywords: Switched Reluctance Machine, Switched Reluctance Drives, Neuro-fuzzy Controller, Parameters of Neuro-fuzzy, Neuro-fuzzy Design.

1 Introduction

Switched Reluctance Machines have been the target in the last decade of an intense interest by researchers particularly solving some problems related with the torque ripple reduction (Kavanagh et al. 1991) with the machine behavior without position sensor (Ertugrul and Cheok 2000), as well as with the machine speed control (Silviano et al. 2003; Kjaer et al. 1995). Our group has presented solutions (Henrique et al. 2000, 2001, 2002; Rodigues et al. 2001; Soares et al. 2001) for these problems with relative success and innovation. Recently, we have developed an experimental speed controller prototype for an 8/6 SRM that uses an on-line learning mechanism to acquire and modify, if needed, the “good” fuzzy control rules.

In this paper, the system power electronics and the switched reluctance machine are briefly introduced. On the other hand, the controller’s tuning and performance is presented by significant experimental tests. These are analyzed and discussed to reveal some advantages of having a learning speed controller for the SR machine, and also the drawbacks that using learning controllers can introduce to the drive system and possible ways to overcome them.

1.1 Motor Type

The SRM used in this work is an 8/6 machine, which means 8 stator poles, with 4 excitation phases, and 6 rotor poles (Miller 1993) presented in Figure 1.1.*d).*

Dealing with the SRM, a finite element modelling analysis has been first effectuated to obtain the magnetic characteristics of the machine, considering the saturation and mutual inductance effects. These results are shown in (Baltazar et al. 2003) where the performance parameters of the machine such as flux linkage per phase, static torque per phase, phase inductance, and the mutual inductances between phases were determined.

1.2 Power Circuit Topology

Figure 1.1.*a)* shows the power converter configuration. It has eight power MOSFETS and eight freewheeling diodes needed to de-energize each phase and protect the switches during the inverse voltage operation. The power converter is supplied by a continuous voltage from a rectifier bridge. This converter topology was explained in detail in (Silviano et al. 2003). It makes possible an individualized phase operation, using three voltage levels (-V, 0 and V), but has the disadvantage of having 2N switches for N motor phases

1.3 Learning Controller

The used neuro-fuzzy system was that originally proposed by (Wang 1992) because of its simplicity, learning capability and the possibility of generating a control law based on rule adaptation (Silviano et al. 2003), minimizing the error goal function. The controller is composed by the blocks presented in Figure 1.1.*e)*, where *ωref* is the speed reference; *ω* is the machine rotor speed and I_{ref} is the reference current. The input variables of the neuro-fuzzy controller are, as usually, the speed error and its variation (e_k,Δe_k). This controller was implemented in a personal computer showed in Figure 1.1.c).

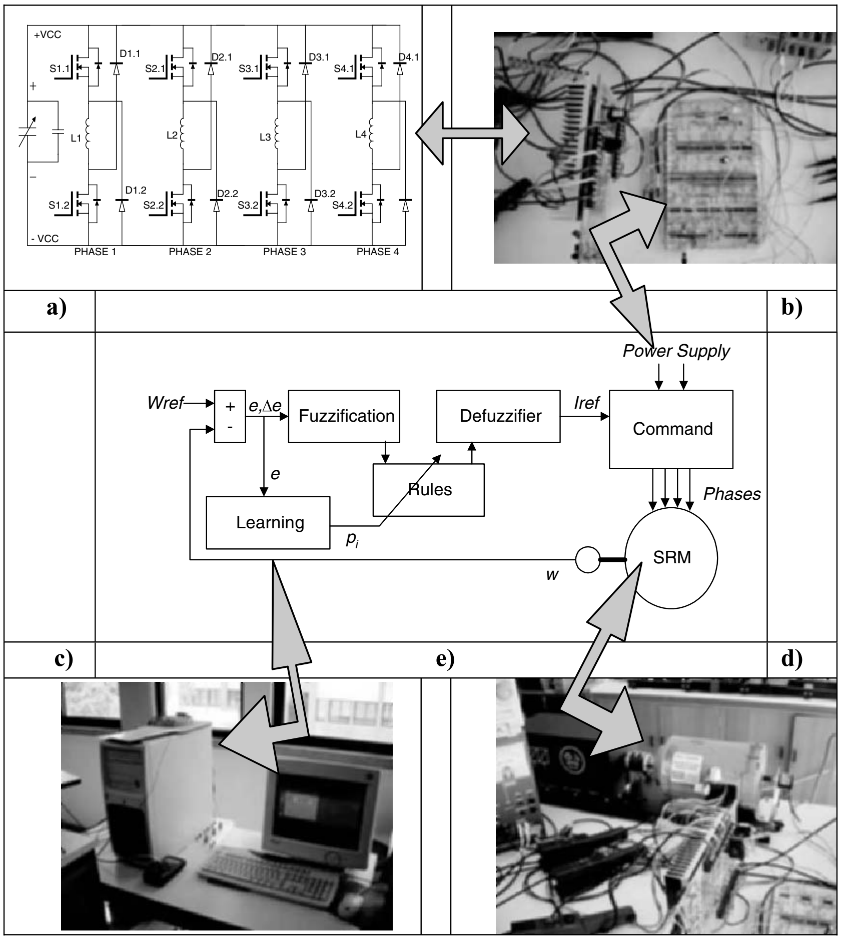


Fig. 1.1. The system controller

2 Neuro-fuzzy Design

This neuro-fuzzy system can be interpreted as a neural network composed with five layers and it is presented in Figure 2.1.*c)*. In the presented model the connections between nodes do not correspond to an attributed weight to the connection but to the propagation of the previous node result.

Each node has an activation function, representative of the fuzzy system for each layer of neurons. Next, the node functions in each of the five layers of this connectionist model are described.

In the first layer, the activation functions are for adjustment of the variables values to the universe of discourse, through a linear function with saturation in the upper/lower limits, as expressed by (1).

O_{1ij} represents the result of the node; the universe of discourse is represented by *U,* where U_{iMAX} and U_{iMIN} are its limits; x_i is the input value where G_{iMAX} and G_{iMIN} are its limits and index *i* is the variable number. In this case the variables will be the speed error and the variation of the error.

$$O_{1,i} = \begin{cases} U_{i\,MAX}; x_i \geq G_{i\,MAX} \\ a_i x_i + b_i; G_{i\,MIN} < x_i < G_{i\,MAX} \\ U_{i\,MIN}; x_i \leq G_{i\,MIN} \end{cases} \; ; i \in \mathrm{N}^+ \tag{1}$$

In the second layer the fuzzification is performed. The output function of each single node is a simple membership function of the fuzzy system. The used membership function is gaussian and it is expressed by (2).

$$O_{2(i,j)} = Exp^{-\frac{1}{2}\left(\frac{O_{1,i}-c_{i,j}}{\mu_{i,j}}\right)} \; ; i \text{ and } j \in \mathrm{N}^+ \tag{2}$$

Where $c_{i,j}$ and $\mu_{i,j}$ are respectively, the gaussian function centre (or mean) and width (or variance) of the jth term of the ith input linguistic variable x_i .

In the third layer the inference mechanism is activated. It is used the *Tnorm*. The used operator is the algebraic product (3).

$$O_{3(i,j)} = O_{2,i} O_{2,j} \; ; i \; and \, j \in \mathrm{N}^+ \tag{3}$$

Where *i* and *j* are the nodes of the second layer associated with the input variable.

The fourth layer performs the consequent part of the rules through the expression (4). The value of the rule weight $\rho_{k(i,\,j)}$ is produced by the fifth layer in function of the learning algorithm.

$$O_{4(i,j)} = O_{3(i,j)} \rho_{K(i,j)} \; ; \; i \; and \, j \in \mathrm{N}^+ \tag{4}$$

The fifth layer has two kinds of nodes. The first one performs the decision signal output. These nodes act as defuzzifier and are expressed in (5).

$$O_{5out} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} O_{4(i,j)}}{\sum_{i=1}^{n}\sum_{j=1}^{n} O_{3(i,j)}} \quad ; \ i \ and \ j \in \mathrm{N}^{+} \tag{5}$$

The second kind of nodes performs the learning in order to minimize the error function (7), represented as an input for correction in the diagram of the proposed fuzzy neural network by modifying the value of the fourth layer nodes through the weighted rules consequents (6).

$$\rho_{K(i,j)} = \rho_{K-1(i,j)} + \gamma \frac{\partial E}{\partial \rho_{K-1(i,j)}} \quad ; i \text{ and } j \in \mathrm{N}^{+} \tag{6}$$

The learning rate γ assumes a value $\in$ [0,1] and E is a cost function to be minimized. E is a quadratic error function (7) where w_{ref} is the value of the speed reference and w_k the value of the machine speed value.

$$E = \frac{1}{2}\left[w_{ref} - w_k\right]^2 \ ; \mathrm{k} \in \mathrm{N}^{+} \tag{7}$$

2.1 Neuro-Fuzzy Parameters

The neuro-fuzzy control system has many parameters for tuning (Arabshahi 1993) such as: learning rate, number and type of membership functions, aggregation of the fuzzification and the weight values of the consequent part of the rules. This is a disadvantage because it does not exist, until then, any formularization or methodology to determine the adjusted values for each situation. The only form used is by trial-and-error until obtaining the best possible performance, quantified by the average quadratic error. In this paper, using the advantage of the accumulated experience with theses systems, the steps that compose this method are presented, as shown in Figure 2.1.*a).*

For diminishing the complexity, some parameters were established at the beginning, in function of the targets, and were considered constant throughout the study of the adjustment process.

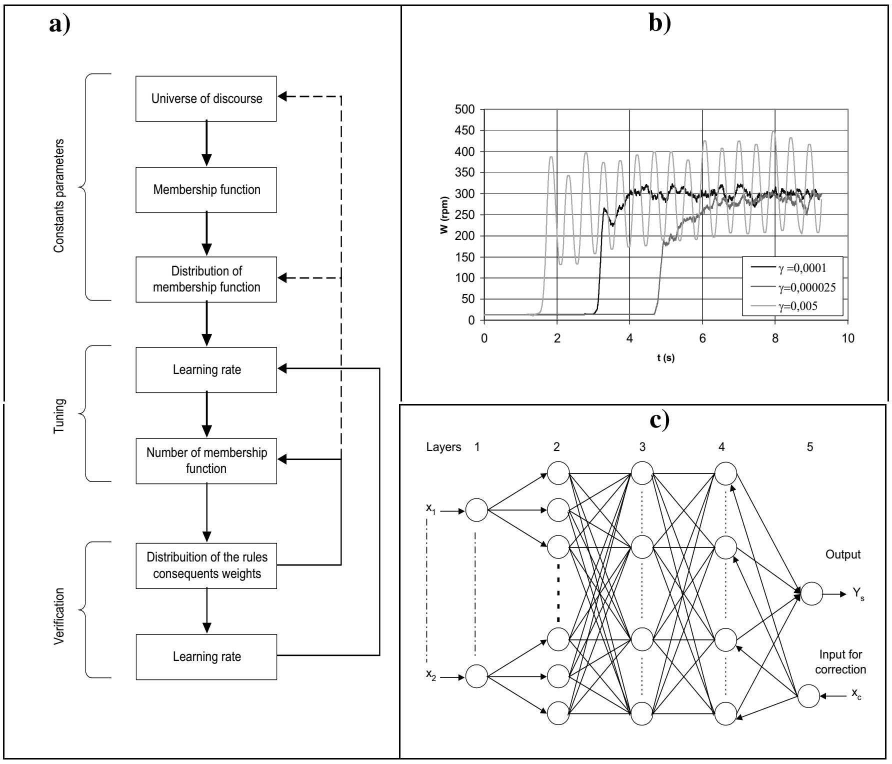


Fig. 2.1 *a)* Blocks diagram of the steps process for tuning the parameters controller *b)* Behaviour of the speed for various learning rates *c)* Diagram of the proposed fuzzy neural network for speed controller

2.1.1 Universe of Discourse

The input and output variables can be normalized, simplifying the universe of discourse. In that case, if it does not intend such, the universe of discourse will be determined in function of its application (Rodrigues 2001)

For example, in this study, the universe of discourse of input variables was defined in function of the intended limits, which were +2000 rpm and -2000 rpm for the speed error and -100 rpm and +100 rpm for the error variation. The output signal is settled between 0 and 30 A, in accordance with the maximum current values of the machine.

2.1.2 Membership Functions

The chosen type of membership functions (2) was the Gaussian type once it is easier its applicability. This one possess two parameters that are the centre $c_{i,j}$ and the variance $\mu_{i,j}$ of the linguistic variable x_i .

The function centre is easily calculated in function of the membership number that cover the universe of discourse.

2.1.3 Distribution of Membership Functions

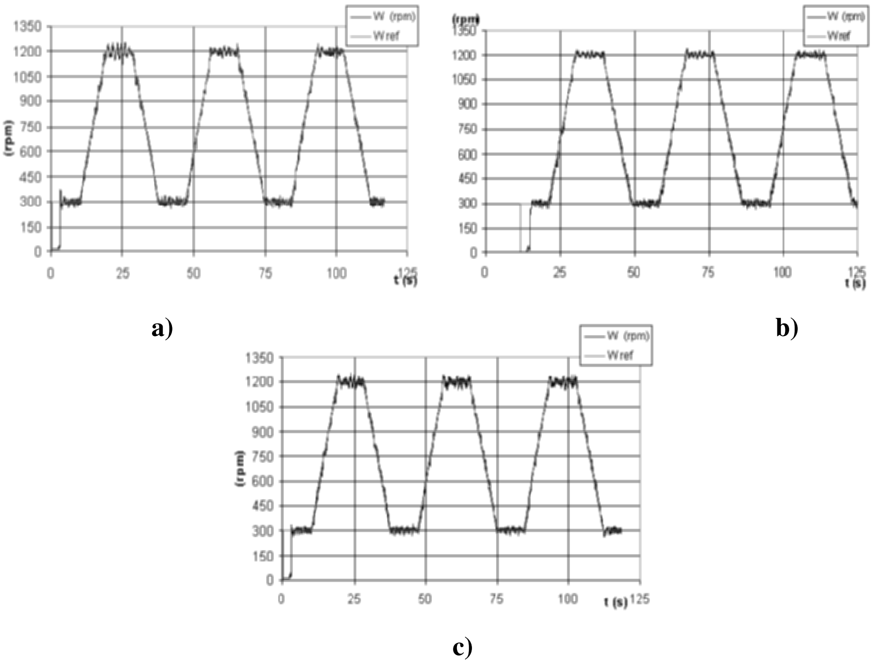
The distribution of the membership functions in the universe of discourse was uniform. The dispersion of theses ones was been determined so that it keeps always the value of μ_i = 0,3 at the intersection point of the membership functions. The parameters being tuned were the learning rate and the number of the membership functions.

2.1.4 Setting the Learning Rate

The experimental tests, with the previously defined parameters were developed only varying the learning rate before a unitary step of +300 rpm were applied in the motor speed reference value. The reason of the learning rate adjustment was to study the reply and the stability of the system behaviour. This test was repeated until obtaining an acceptable learning rate value. This elapsed time situates between the instant where the system is activated until the machine starts running. This time interval is going bigger as gamma becomes smaller (γ = 0,0001 t =3,15 s; γ = 0,000025 t = 4,86 s; γ = 0,005 t =1,75 s). Figure 2.1.*b)* also discloses the influence of the learning rate for the system stability. That is why the selected learning rate value selected was 0,0001 for this test. The numbers of membership functions were not very important, in this global experience, since they cover the totality of the universe of discourse.

2.1.5 Number of Membership Functions

Next step was, keeping the values of the already established constant parameters, studying the system behaviour by varying the number of membership functions. The same number of membership functions was applied to the two input variables. In Figure 2.2.*a)*, *b)* and *c)* are presented the experimental results of the system behaviour of the system with 13, 15 and 17 membership functions, respectively, being the speed reference signal a trapezoidal function. In Figure 2.2.*a)* an initial oscillatory is observed that however converges reasonably.



a)

b)

c)

Fig. 2.2. Speed Tracking with a trapezoidal reference function *a)* with 13 membership functions, *b)* with 15 membership functions *c)* with 17 membership functions

Through the figures analysis it is easy to perceive that there are differences between the signal and the reference, which can be quantified by the MPE factor (mean percentage error) and the MAPE factor (mean absolute percentage error), expressed by (9) and (10) respectively, with i ∈ N+.

$$MPE = \sum_{i=1}^{n} \frac{PE_i}{n} \tag{9}$$

$$MAPE = \sum_{i=1}^{n} \frac{|PE_i|}{n} \tag{10}$$

Where PE is the error percentage of the speed given by (11)

$$PE_i = \frac{(\varpi - \varpi ref)}{\varpi ref} 100 \tag{11}$$

To evaluate the performance of the neuro-fuzzy system with diverse fuzzy structures, the values of MAPE and MPE of the motor system cycles displayed in Figures 2.2.*a)*, *b)* and *c)* are shown in Table 2.1. Table 2.1. presents a global behaviour analysis and a second and third cycle analysis too. For all these tests the consequent values of the rules were initialized with zero value. It means that, in the first cycle, the adaptation to the reference signal is made and it is in this cycle that the trajectory is covered for the first time.

Table 2.1. Comparative table of *MAPE* and *MPE* of the neuro-fuzzy system with diverse membership functions

Membership functions	Global		2º Cycle		3º Cycle	
	MAPE [%]	MPE [%]	MAPE [%]	MPE [%]	MAPE [%]	MPE [%]
11	4,8	-2,15	2,55	-0,072	2,53	-0,044
13	4,3	-1,61	2,46	-0,026	2,49	0,004
15	4,2	-1,43	2,63	0,023	2,63	-0,094
17	4,9	-2,13	2,85	-0,026	2,74	0,003
19	5,9	-2,40	3,5	0,023	3,59	0,105

The analysis of Table 2.1 makes evident that the better MAPE global performance controller occurs for 15 membership functions. The controller shows also a good partial performance with 13 membership functions. It is evident that depending on the application it will always need for a commitment solution. If the drive system is in steady-state with few changes and if the time sampling is not important, in terms of bad contribution for the learning mechanism, then the configuration with 13 membership functions is satisfactory. On the contrary the system with 15 membership functions has a little better performance and so it is used in this study. After tuning the number of membership functions it will be shown the verification tests.

3 Verification Tests

The verification tests have the purpose of supplying information that, conjugated with the existing ones presented in the previous sections, allow to achieve the fine adjustment of some parameters.

3.1 Weights Distribution

The surface representation of the weights matrix of the rules consequents and their distribution allow, in this case, to detect imperfections or faults in the universe of discourse, number of membership functions and its spreading. For example, an insufficient universe of discourse is observed by the rise of the weights value in the edges. Excess in the universe of discourse amplitude is detected by a very intent rise in the center of the weights matrix. Depending on the intended one, it will be able not only to correct the universe of discourse, as the number of membership functions as its distribution.

3.2 Learning Rate

In this verification it is intended to do the fine tuning of the learning rate value. In this example it is used previously the same reference trapezoidal function with diverse learning rate values. The purpose is to find one that iteratively allows to improve the set performance. In Figure 3.1 is presented the speed tracking with a trapezoidal reference and various learning rates like 0,0001, 0,0002, 0,0003 and 0,0004 in a second, third, fourth and fifth cycle, respectively.

Table 3.1 presents the measure of the error applied to each chunk of the trapezoidal function. A tracking cycle of the reference was divided in four segments. Theses segments are the two speed platforms of 300 rpm and 1200 rpm and the two slopes of ascending and descending speed. Thus, they all have a better notion of the system behaviour in each segment presented in Table 3.1. In this table, the best value of learning rate for the platforms of 300 rpm and 1200 rpm is 0,0001. For the speed ascending slope the best learning rate is 0,0002 and for the descendant slope the learning rate would be 0,0003. Thus, one concludes that the best global performance would be achieved through the composition of learning rates depending on the situation. In the case that it is not possible, and for the steady state situations the best learning rate was applied, or either 0,0001.

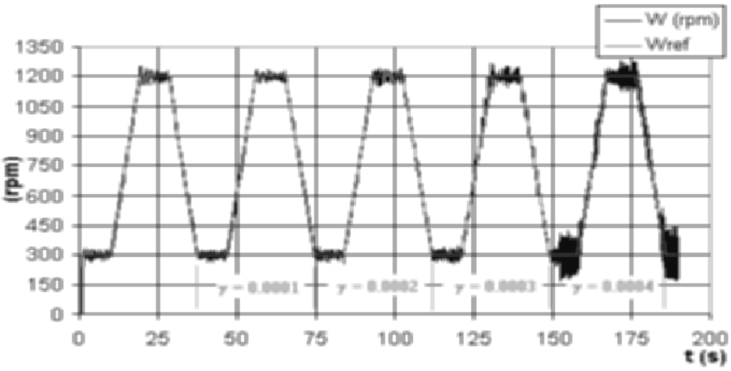


Fig. 3.1. Speed tracking with trapezoidal reference function and various learning rate values

Table 3.1. Comparative table of MAPE and MPE of the neuro-fuzzy system with diverse learning rate values.

[%]	300 [rpm]		ω ascendant		1200 [rpm]		ω descendant	
γ	MAPE [%]	MPE [%]	MAPE [%]	MPE [%]	MAPE [%]	MPE [%]	MAPE [%]	MPE [%]
0.0001	2,990	-0,044	3,484	-3,429	0,843	0,043	3,383	3,257
0.0002	3,189	0,166	2,374	-1,815	1,300	-0,014	2,376	1,741
0.0003	4,409	-0,057	3,011	-1,309	1,699	0,085	2,304	1,036
0.0004	15,104	-0,061	5,637	-0,818	2,204	0,023	6,725	0,730

4 Experimental Results

The laboratorial tests were set up by tracking a trapezoidal speed reference function between 300 rpm and 1200 rpm, 500 rpm and 1400 rpm and finally between 400 rpm and 800 rpm, as presented in Figure 4.1. The slope of the first and second group of speed cycles has the angular acceleration of 1,618 rad/s2 and in the last cycle is 0,719 rad/s2.

The error is presented by a set of cycles of speed levels in Table 4.1.. It is evident that the performance presented in the interval between 400 rpm and 800 rpm is the best. This is due to the fact that the acceleration being lower implies a better adaptation of the neuro-fuzzy system to the speed reference signal tracking.

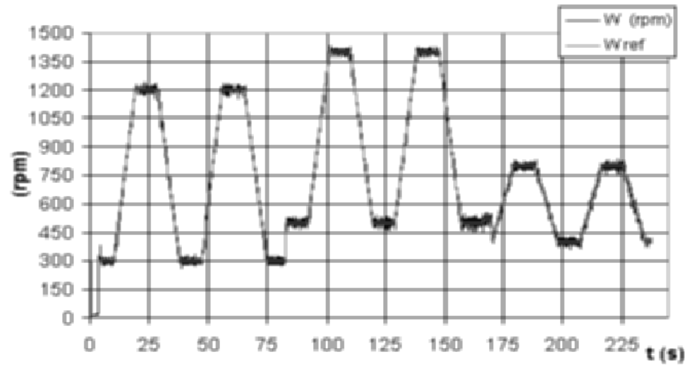


Fig. 4.1. Speed tracking with different speed level

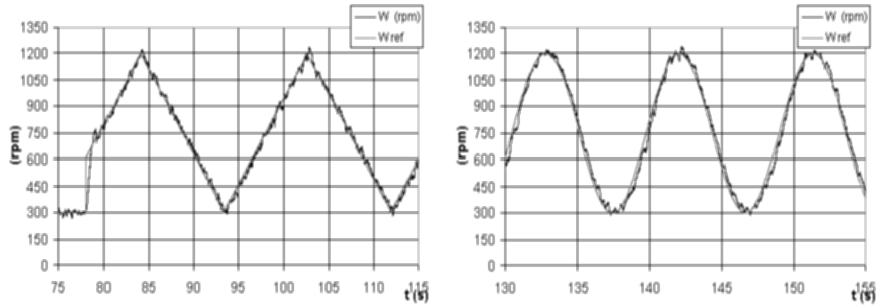
Table 4.1. Error performance analysis by speed cycles

Cycles	300 / 1200 [rpm]	500 / 1400 [rpm]	400 / 800 [rpm]
MAPE [%]	2,94	2,269	2,254
MPE [%]	0,0051	0,1292	0,1458

The tracking of the triangular speed reference function is presented in Figure 4.2.*a)*.

The speed tracking error is 0,198% for MPE and 3,0% for MAPE. It is slightly higher than the one presented with a trapezoidal function in Table 4.1..

The tracking of a sinusoidal speed reference function is presented in Figure 4.2.*b)*. The error with MPE is 0,325% and with the MAPE is 5,131%, assuming higher values related to the previous ones, due to the increase of the slope. The acceleration for the triangular and trapezoidal functions is 1,618 rad/s2 and for the sinusoidal function is 4,592r/s2 which represents an increase of 283,8%.



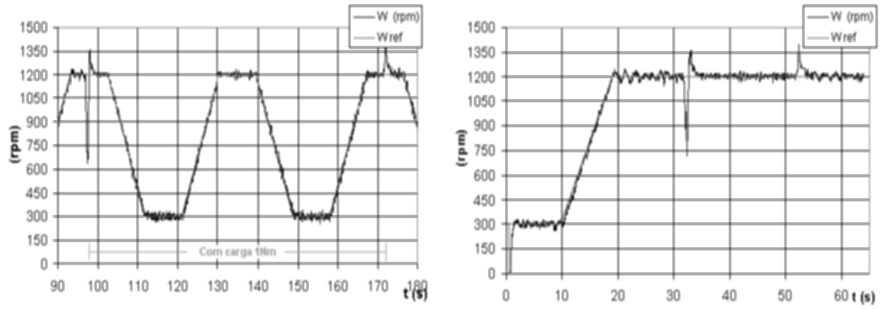
a)

b)

Fig. 4.2. Speed tracking with a) triangular reference function b) sinusoidal reference function

The laboratorial test in tracking a trapezoidal reference function was followed by applying a mechanical load of 1Nm during two cycles. The load was applied and withdrawal brusquely. The speed variation is presented in Figure 4.3.*a)*.

In this situation it is evident a reduction of the absolute average error value (MAPE 2.59%) relatively to the tests without mechanical load (MAPE 2.94%).



a)

b)

Fig. 4.3. Speed tracking with *a)* a trapezoidal reference function with 1 Nm mechanical load *b)* Constant speed with mechanical load

In the controller point of view, an unfavourable situation is when the system does not have a connected mechanical load. Another unfavourable situation is when the phase currents have overlapping, which implies torque oscillations that will be reflected in the machine speed. The load attenuates these oscillations in such a way that it is reflected in the MAPE value.

To analyse this situation a mechanical load is applied when the machine speed is constant, as can be seen in Figure 4.3.*b)*. It can also be seen the two speed perturbations associated with the application of a resistant torque. In the first one, the speed suddenly decays, being the recovery dependent on the reply of the learning algorithm. This also happens when the load is removed, appearing the rise of the speed. The values of the MAPE and the MPE were calculated and presented in Table 4. These values are lower with load than without load.

The load variation does not influence the machine speed if the variation rate is slower than the cadence of the rules time update by the learning algorithm.

Table 4.2. Performance analysis of the error by speed cycles

	1200 rpm without load	1200 rpm with load
MAPE [%]	0,877	0,673
MPE [%]	0,0473	0,0217

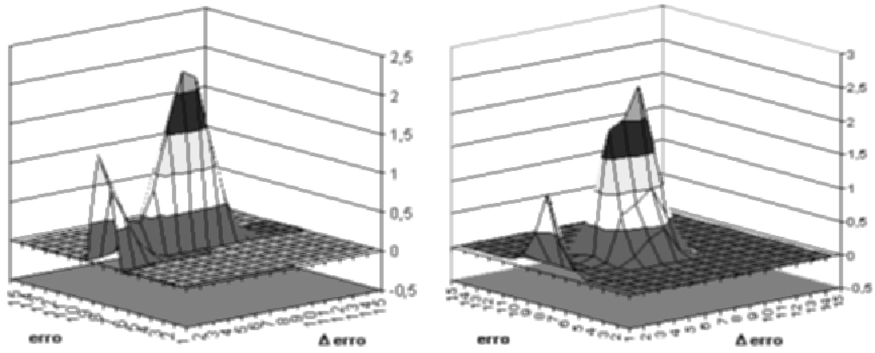
4.1 Control Surface

The system behaviour is now analyzed changing the rules weight matrix. This matrix is also known as control surface, and is formed by the weight values of the consequent one of each rule. All the experimental tests, whose results will be presented, were always initiated with a null matrix of the rules. In Figure 4.4.*a)* it is presented the control surfaces when the speed level is 300 rpm and with a trapezoidal reference speed function. In Figure 4.4. b) it is presented the control surfaces when the speed level is 1200 rpm and the neuro-fuzzy system is composed by 15 membership functions (for the error and error variation).

In Figure 4.4.*a)*, with reference speed of 300 rpm, the lower peak zone is when the variation of the error is strongly negative and happens when there is a sudden variation of speed. The central part (higher peak zone) is when it comes close to the reference value. As more narrow will be the base, of the control surface with a conic geometric figure, better will be the trajectory convergence to the intended value. The goal is to reach the error zero value and the error variation zero value.

The same happens in Figure 4.4.*b)*, with the difference that the reference speed varies between 300 rpm and 1200 rpm. The covered trajectory was such that it resulted in a central geometric figure of control surface with a wider base and different faces with different slopes and levels compared with the previous one.

Once the rules matrix is always the same one, it implies that, according to the evolution of the error, the learning algorithm will be always used to modify and to adjust the control surface. This constant adaptation of the control surface becomes, according to the amplitude, one determined time interval.



a) **b)**

Fig. 4.4. Surface control *a)* 300 rpm *b)* 1200 rpm

The reference speed variations should be, in terms of time, higher than the sampling time and higher than the time of adaptation of the rules matrix, in order to guarantee an acceptable accompaniment.

4.2 PID Controller

In the most generic industrial applications the PID controller is one of the most used and popular controller. The expression of the discrete PID approximation is presented in (12) and it was applied with a digital control with periodic update of *Ts* equal to 2 ms. This update time is the cycling period of the processing algorithm in the PC based implementation. The term e_k is the error and Δe_k the variation of the error. PID controllers may be tuned in a variety of ways (Aström and Wittenmark,1993) K_p - proportional gain and K_i - integral gain are tuned by hand tuning until it is observed the best possible performance. The advantage of hand tuning is that uses on-line procedure and develops a feeling on how the system behaves in a closed loop. A disadvantage is that it takes a long time to develop this feeling, and it is difficult to sense whether the final settings are optimal.

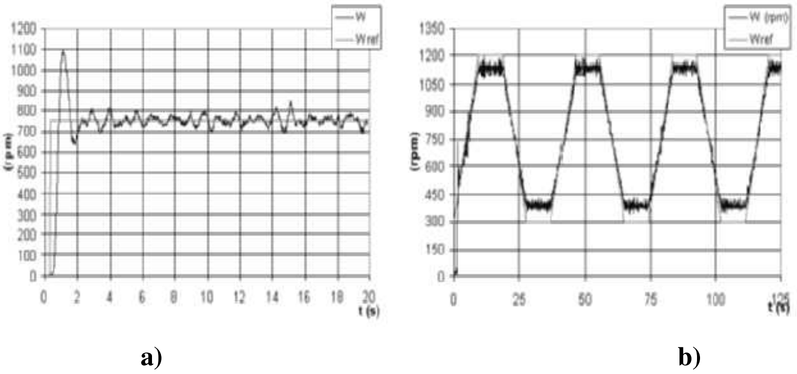
$$I_{ref_k} = K_p \left(e_k + K_i \sum e_k T_s + K_d \frac{\Delta e_k}{T_s} \right) \tag{12}$$

The gains were adjusted in order to get an acceptable performance in the regimen of 750 rpm of speed. The course of the speed is presented in Figure 4.5.*a)*.

The MAPE and the MPE of the speed error obtained from Figure 4.5.*a)* is 2,621% and 0,658% respectively. These are the order of magnitude of the gotten values with the neuro-fuzzy controller.

However when the reference is the already used trapezoidal function, it is clear that the error significantly increases, as can be seen in Figure 4.5.*b)*. One could think about the possibility of the existence of an offset if the shunting line was only in relation with a speed level, but this is not what occurs. It is verified that close to a functioning regimen, it is possible to adjust the PID controller for this point, until it presents comparable errors with the ones of the neuro-fuzzy system. However, when it is moved away from this functioning point, the errors increase considerable.

It is verified that it would be necessary to readjust the K_p, K_i and K_d parameters of the PID controller for each functioning regimen of the machine.



a) **b)**

Fig. 4.5. *a)* Constant speed with PID controller *b)* Speed tracking with a trapezoidal reference function (MAPE>5%)

5 Conclusions

A neuro-fuzzy system for SRM was designed and implemented. All the five layers of this connectionist model are explained in this paper. It is presented the learning algorithm in order to minimize the error function, which acts modifying the values of the weighted rules consequents. The controller parameters are defined and explained. Some parameters were fixed while others vary until satisfactory system behaviour is reached.

Experimental results of the system performance are presented along a set of various tracking speed reference functions. In this work it was used two kinds of error measures analysis, the MPE and MAPE. One may conclude that the performance behaviour in the speed tracking of a trapezoidal reference function is better when the angular acceleration is lower. This is due to a better adaptation of the neuro-fuzzy system to the speed reference. It is also presented a tracking of a speed reference function with a mechanical load. It is observed an improvement in the speed behaviour performance of the system.

The surfaces controls generated by the matrix weights consequents rules of the neuro-fuzzy system for two distinct situations are also shown.

Finally, it is presented a PID controller with hand tuned parameters for a specific machine regimen and tested with a tracking reference speed trapezoidal function, which reveals the systems necessity for an adaptation capability.

It is not the main goal of this work to determine the new PID controller gain values but it was demonstrated that the SRM control system will need to possess an adaptation capability.

References

Arabshahi P, RJ Marks II and Caudell TP (1993) Adaptation of Fuzzy Inferencing: a survey. Proceedings of the IEEE/Nagoya University WWW on Learning and Adaptive Systems, Nagoya University, pp 1-9.

Aström, KJ and Wittenmark, Computer controlled systems theory and design. Prentice-Hall, Englewood Cliffs.

Baltazar P, Silviano R, Pires AJ and Costa Branco PJ (2003) Obtaining the Magnetic Characteristics of an 8/6-Switched Reluctance Machine: FEM Analysis and Experimental Tests. IEEE International Symposium on Industrial Electronics - ISIE2003,Rio de Janeiro.

Henriques L, Costa Branco PJ, Rolim L, Suemitsu W and Dente J (2000) Torque Ripple Minimization in a Switched Reluctance Drive by Neuro-Fuzzy Compensation. IEEE Transactions on Magnetics, Vol 36, N 5/part 1, pp 3592-3594.

Henriques L, Costa Branco PJ, Rolim L, Suemitsu W (2001) Two Automatic Online New Schemes to Compensate the Torque Ripple of Switched Reluctance Machines: With and Without Torque Signal Measurement. Soft Computing And Industry – Recent Applications, Springer-Verlag , Berlim, pp. 225-236.

Henriques L, Costa Branco PJ, Rolim L, Suemitsu W (2002) Proposition of an Off-Learning Current Modulation for Torque Ripple Reduction in Switched Reluctance Motors: Design and Experimental Evaluation. IEEE Transactions on Industrial Electronics, Vol 49, N 3, pp 665-676.

Henriques L, Rolim L, Suemitsu W, Costa Branco PJ (2004) Development and Implementation of a Neuro Fuzzy Technique for position sensor elimination in a SRM. International Symposium on Industrial Electronics, Ajaccio, France.

Miller TJE (1993) Switched Reluctance Motors and Their Control. Magna Phisics Publ. and Clarendon Press, Oxford.

Rodrigues M, Costa.Branco PJ, Suemitsu W and Dente J (2001) Fuzzy-Logic Torque Ripple Reduction by Turn-off Angle Compensation for a Switched Reluctance Motor. IEEE Transactions On Industrial Electronics, Vol 48, N 3, pp 711-715.

Ertugrul N and Cheok AD (2000) Indirect angle Estimation in Switched Reluctance Motor Drives using Fuzzy Logic Based Motor Model. IEEE Transactions on Power Electronics Vol 15, N 6, pp 1029-1044.

Silviano R, Pires AJ and Costa Branco PJ (2003) Implementation of an 8/6 Switched Reluctance Mosfet Current Controller: Simulation Study And Experimental Tests. IEEE International Symposium on Industrial Electronics (ISIE'2003), Rio de Janeiro, Brasil. ISBN 0-7803-7912-8

Silviano R, Pires AJ and Costa Branco PJ (2003) Implementation of a Neuro-Fuzzy Speed Controller for a Switched Reluctance Machine. 8º CLEEE-Congresso Luso-Espanhol de Engenharia Electrotécnica, Vol 3, pp 6.345-6.350.

Soares F and Costa Branco PJ (2001) Simulation of a 6/4 Switched Reluctance Motor Based on Matlab/Simulink Environment. IEEE Transactions on Aerospace and Electronic Systems, Vol 37, N 3, pp 989-1099.

Wang LX and Mendel JM (1992) Generating Fuzzy Rules by Learning from Examples. IEEE Transaction on Systems Man Cybernetics, Vol 22, N 6, pp 116-132, 1414-1427.

Part XV

Hybrid Intelligent Systems using Fuzzy Logic, Neural Networks and Genetic Algorithms

Modular Neural Networks and Fuzzy Sugeno Integral for Face and Fingerprint Recognition

Patricia Melin, Claudia Gonzalez, Diana Bravo, Felma Gonzalez and Gabriela Martinez

Dept. of Computer Science, Tijuana Institute of Technology, Tijuana, Mexico

Abstract-We describe in this paper a new approach for pattern recognition using modular neural networks with a fuzzy logic method for response integration. We proposed a new architecture for modular neural networks for achieving pattern recognition in the particular case of human faces and fingerprints. Also, the method for achieving response integration is based on the fuzzy Sugeno integral with some modifications. Response integration is required to combine the outputs of all the modules in the modular network. We have applied the new approach for fingerprint and face recognition with a real database from students of our institution.

1 Introduction

Response integration methods for modular neural networks that have been studied, to the moment, do not solve well real recognition problems with large sets of data or in other cases reduce the final output to the result of only one module. Also, in the particular case of face recognition, methods of weighted statistical average do not work well due to the nature of the face recognition problem. For these reasons, a new approach for face and fingerprint recognition using modular neural networks and fuzzy integration of responses was proposed in this paper.

The basic idea of the new approach is to divide a human face into three different regions: the eyes, the nose and the mouth, and the fingerprint also into three parts, top, middle and bottom. Each of these regions is assigned to one module of the neural network. In this way, the modular neural network has three different modules, one for each of the regions of the human face and the fingerprint. At the end, the final decision of face and fingerprint recognition is done by an integration module, which has to take into account the results of each of the modules. In our approach, the integration module uses the fuzzy Sugeno integral to combine the outputs of the three modules. The fuzzy Sugeno integral allows the integration of responses from the three modules of the eyes, nose and mouth of a human specific face and the integration of the responses from the three modules of the fingerprint parts. Other approaches in the literature use other types of integration modules, like voting methods, majority methods, and neural networks.

The new approach for face and fingerprint recognition was tested with a database of students and professors from our institution. This database was collected at our institution using a digital camera for the faces and a special scanner for the fingerprints. The results with our new approach for face and fingerprint recognition on this database were excellent.

2 Modular Neural Networks

There exists a lot of neural network architectures in the literature that work well when the number of inputs is relatively small, but when the complexity of the problem grows or the number of inputs increases, their performance decreases very quickly. For this reason, there has also been research work in compensating in some way the problems in learning of a single neural network over high dimensional spaces.

In the work of Sharkey (Sharkey 1998), the use of multiple neural systems (Multi-Nets) is described. It is claimed that multi-nets have better performance or even solve problems that monolithic neural networks are not able to solve. It is also claimed that multi-nets or modular systems have also the advantage of being easier to understand or modify, if necessary.

In the literature there is also mention of the terms “ensemble” and “modular” for this type of neural network. The term “ensemble” is used when a redundant set of neural networks is utilized, as described in Hansen and Salomon (Hansen and Salomon 1990). In this case, each of the neural networks is redundant because it is providing a solution for the same task, as it is shown in Figure 1.

On the other hand, in the modular approach, one task or problem is decompose in subtasks, and the complete solution requires the contribution of all the modules, as it is shown in Figure 2.

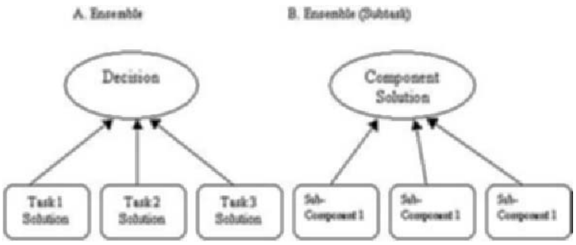


Fig. 1. Ensembles for one task and subtask.

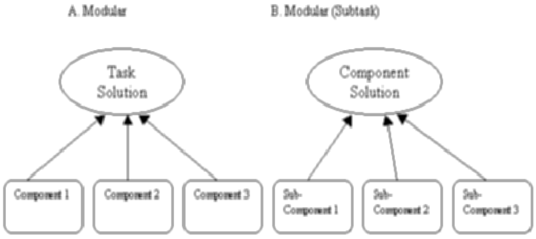


Fig. 2. Modular approach for task and subtask.

2.1 Multiple Neural Networks

In this approach we can find networks that use strongly separated architectures. Each neural network works independently in its own domain. Each of the neural networks is build and trained for a specific task. The final decision is based on the results of the individual networks, called agents or experts. One example of this decision is shown by (Albrecht 1996), as shown in Figure 3, where a multiple architecture is used, one module consists of a neural network trained for recognizing a person by the voice, while the other module is a neural network trained for recognizing a person by the image.

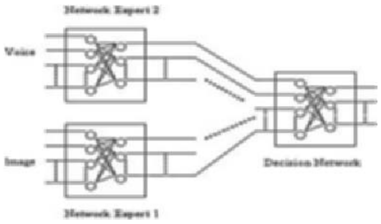


Fig. 3. Multiple networks for voice and image.

The outputs by the experts are the inputs to the decision network, which is the one making the decision based on the outputs of the expert networks.

2.2 Main Architectures with Multiple Networks

Within multiple neural networks we can find three main classes of this type of networks (Fu et al. 2001):

- Mixture of Experts (ME): The mixture of experts can be viewed as a modular version of the multi-layer networks with supervised training or the associative version of competitive learning. In this design, the local experts are trained with the data sets to mitigate weight interference from one expert to the other.
- Gate of Experts: In this case, an optimization algorithm is used for the gating network, to combine the outputs from the experts.
- Hierarchical Mixture of Experts: In this architecture, the individual outputs from the experts are combined with several gating networks in a hierarchical way.

2.3 “Modular”Neural Networks

The term “Modular Neural Networks” is very fuzzy. It is used in a lot of ways and with different structures. Everything that is not monolithic is said to be modular. In the research work by (Boers and Kuiper 1992), the concept of a modular architecture is introduced as the development of a large network using modules.

One of the main ideas of this approach is presented in (Albrecht 1996), where all the modules are neural networks. The architecture of a single module is simpler and smaller than the one of a monolithic network. The tasks are modified in such a way that training a subtask is easier than training the complete task. Once all modules are trained, they are connected in a network of modules, instead of using a network of neurons. The modules are independent to some extent, which allows working in parallel. Another idea about modular networks is presented by (Boers and Kuiper 1992), where they used an approach of networks not totally connected. In this model, the structure is more difficult to analyze, as shown in Figure 4. A clear separation between modules can’t be made. Each module is viewed as a part of the network totally connected.

In this figure, we can appreciate two different sections from the monolithic neural network, namely A and B. Since there are no connections between both parts of the network, the dimensionality (number of weights) is reduced. As a consequence the required computations are decreased and speed of convergence is increased.

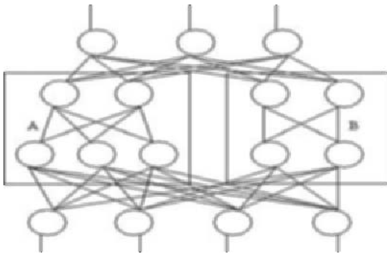


Fig. 4. One type of modular neural network.

2.4 Advantages of Modular Neural Networks

A list of advantages of modular networks is given below:

- They give a significant improvement in the learning capabilities, over monolithic neural networks, due to the constraints imposed on the modular topology.
- They allow complex behavior modeling, by using different types of knowledge, which is not possible without using modularity.
- Modularity may imply reduction of number of parameters, which will allow and increase in computing speed and better generalization capabilities.
- They avoid the interference that affects "global" neural networks.
- They help determine the activity that is being done in each part of the system, helping to understand the role that each network plays within the complete system.
- If there are changes in the environment, modular networks enable changes in an easier way, since there is no need to modify the whole system, only the modules that are affected by this change.

2.5 Elements of Modular Neural Networks

When considering modular networks to solve a problem, one has to take into account the following points (Ronco and Gawthhrop 1995):

- Decompose the main problem into subtasks.
- Organizing the modular architecture, taking into account the nature of each subtask.
- Communication between modules is important, not only in the input of the system but also in the response integration.

In the particular case of this paper, we will concentrate in more detail in the third point, the communication between modules, more specifically information fusion at the integrating module to generate the output of the complete modular system.

2.6 Main Task Decomposition into Subtasks

Task Decomposition can be performed in three different ways, as mentioned by (Lu and Ito 1998):

□ Explicit Decomposition: In this case, decomposition is made before learning and requires that the designer has deep knowledge about the problem. Of course, this maybe a limitation if there isn't sufficient knowledge about the problem.

□ Automatic Decomposition: In this case, decomposition is made as learning is progressing.

□ Decomposition into Classes: This type of decomposition is made before learning, a problem is divided into a set of sub-problems according to the intrinsic relations between the training data. This method only requires knowledge about the relations between classes.

2.7 Communication Between Modules

In the research studies made by (Ronco and Gawthrop 1995), several ways of achieving communication between modules are proposed. We can summarize their work by mentioning the following critical points:

1. How to divide information, during the training phase, between the different modules of the system.
2. How to integrate the different outputs given by the different modules of the system to generate the final output of the complete system.

2.8 Response Integration

Response integration has been considered in several ways, as described by (Smith and Johansen 1997) and we can give the following list:

□ Using Kohonen's self organizing maps, Gaussian mixtures, etc.

□ The method of "Winner Takes All", for problems that require similar tasks.

□ Models in series, the output of one module is the input to the following one.

- Voting methods, for example the use of the “Softmax” function.
- Linear combination of output results.
- Using discrete logic.
- Using finite state automata.
- Using statistical methods.
- Using fuzzy logic (Castillo and Melin 2003).

3 Methods of Response Integration

The importance of this part of the architecture for pattern recognition is due to the high dimensionality of this type of problems. As a consequence in pattern recognition is good alternative to consider a modular approach. This has the advantage of reducing the time required of learning and it also increases accuracy. In our case, we consider dividing the images of a human face in three different regions. We also divide the fingerprint into three parts, and applying a modular structure for achieving pattern recognition.

In the literature we can find several methods for response integration, that have been researched extensively, which in many cases are based on statistical decision methods. We will mention briefly some of these methods of response integration, in particular the ones based on fuzzy logic. The idea of using these types of methods, is that the final decision takes into account all of the different kinds of information available about the human face and fingerprint. In particular, we consider aggregation operators, and the fuzzy Sugeno integral (Sugeno 1974).

Yager (1999) mentions in his work, that fuzzy measures for the aggregation criteria of two important classes of problems. In the first type of problems, we have a set Z={z1,z2,…,zn} of objects, and it is desired to select one or more of these objects based on the satisfaction of certain criteria. In this case, for each zi∈Z, it is evaluated D(zi)=G(Ai(zi),…,Aj(zi)), and then an object or objects are selected based on the value of G. The problems that fall within this structure are the multi-criteria decision problems, search in databases and retrieving of documents.

In the second type of problems, we have a set G={G1,G2,…,Gq} of aggregation functions and object z. Here, each Gk corresponds to different possible identifications of object z, and our goal is to find out the correct identification of z. For achieving this, for each aggregation function G, we obtain a result for each z, Dk(z)=Gk(A1(z), A2(z), … ,An(z)). Then we associate to z the identification corresponding to the larger value of the aggregation function.

A typical example of this type of problems is pattern recognition. Where Aj corresponds to the attributes and Aj(z) measures the compatibility of z with the attribute. Medical applications and fault diagnosis fall into this type of problems. In diagnostic problems, the Aj corresponds to symptoms associated with a particular fault, and Gk captures the relations between these faults.

3.1 Fuzzy Integral and Fuzzy Measures

Fuzzy integrals can be viewed as non-linear functions defined with respect to fuzzy measures. In particular, the "gλ-fuzzy measure" introduced by (Sugeno 1974) can be used to define fuzzy integrals. The ability of fuzzy integrals to combine the results of multiple information sources has been mentioned in previous works.

Definition 1. A function of sets g:2x-(0.1) is called a fuzzy measure if:

1. 1) g(0)=0 g(x)=1
2. 2) g(A)≤ g(B) if A⊂B
3. 3) if {Ai}iα =1 is a sequence of increments of the measurable set then

$$\lim_{i \to \infty} g(Ai) = g\ (\lim_{i \to \infty} Ai) \qquad (1)$$

From the above it can be deduced that g is not necessarily additive, this property is replaced by the additive property of the conventional measure.

From the general definition of the fuzzy measure, Sugeno introduced what is called "gλ-fuzzy measure", which satisfies the following additive property:

For every A, B ⊂ X and A ∩ B = θ,

$$g(A \cup B) = g(A) + g(B) + \lambda\ g(A)g(B), \qquad (2)$$

for some value of λ>-1.

This property says that the measure of the union of two disjunct sets can be obtained directly from the individual measures. Using the concept of fuzzy measures, (Sugeno 1974) developed the concept of fuzzy integrals, which are non-linear functions defined with respect to fuzzy measures like the gλ-fuzzy measure.

Definition 2. let X be a finite set and h:X→[0,1] be a fuzzy subset of X, the fuzzy integral over X of function h with respect to the fuzzy measure g is defined in the following way,

$$h(x) \text{ o } g\ (x) = \max_{E \subseteq X} [\ \min (\ \min_{x \in E} h(x), g(E))] \qquad (3)$$

$$= \sup\ [\min(\ \alpha\ ,\ g(h\alpha\ ))]\ \alpha \in [0,\ 1]$$

where $h\alpha$ is the level set α of h,

$$h\alpha = \{ x \mid h(x) \geq \alpha \}. \qquad (4)$$

We will explain in more detail the above definition: h(x) measures the degree to which concept h is satisfied by x. The term min(hx) measures the degree to which concept h is satisfied by all the elements in E. The value g(E) is the degree to which the subset of objects E satifies the concept measure by g. As a consequence, the obtained value of comparing these two quantities in terms of operator min indicates the degree to which E satifies both criteria g and min(hx). Finally, operator max takes the greatest of these terms. One can interpret fuzzy integrals as finding the maximum degree of similarity between the objective and expected value.

4 Proposed Architecture and Results

In the experiments performed in this research work, we used 20 photographs that were taken with a digital camera and 20 fingerprints from students and professors of our Institution. The photographs were taken in such a way that they had 148 pixels wide and 90 pixels high, with a resolution of 300x300 ppi, and with a color representation of a gray scale, some of these photographs are shown in Figure 5. In addition to the training data (20 photos) we did use 10 photographs that were obtained by applying noise in a random fashion, which was increased from 10 to 100%.

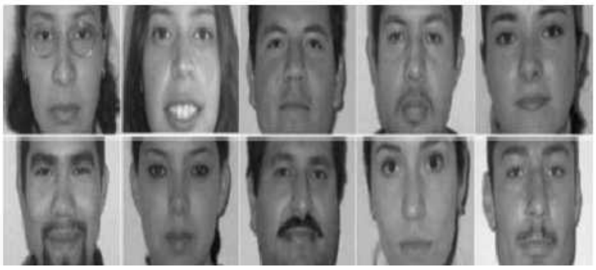


Fig. 5. Sample Faces Used for Training.

The images of fingerprints (Quezada 2004) were taken in such a way that they had 198 pixels wide and 200 pixels high, with a resolution of 300x300 ppi, and with a color representation of a gray scale, some of these images are shown in Figure 6. In addition to the training data (20 fingerprints) we did use 10 images that were obtained by applying noise in a random fashion, which was increased from 10 to 100%.

Fig. 6. Sample fingerprints used for training.

4.2 Proposed Architecture

The architecture proposed in this work consist of three main modules, in which each of them in turn consists of a set of neural networks trained with the same data, which provides the modular architecture shown in Figure 7.

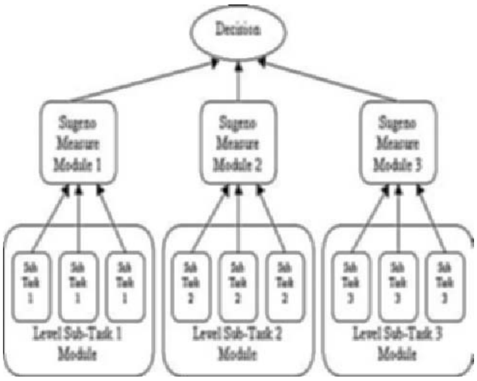


Fig. 7. Final proposed architecture.

The input to the modular system is a complete photograph. For performing the neural network training, the images of the human faces were divided in three different regions. The first region consists of the area around the eyes, which corresponds to Sub Task 1. The second region consists of the area around the nose, which corresponds to Sub Task 2. The third region consists of the area around the mouth, which corresponds to Sub Task 3. An example of this image division is shown in Figure 8.

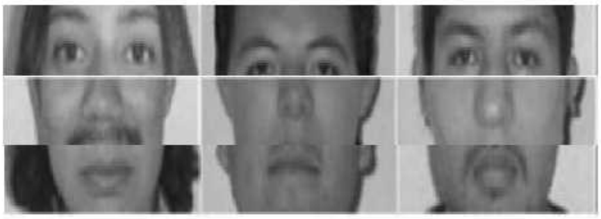


Fig. 8. Example of Image Division.

As output to the system we have an image that corresponds to the complete image that was originally given as input to the modular system, we show in Figure 9 an example of this for face recognition. In the same way the fingerprints are divided in three parts and given to the corresponding Sub task module. This is ilustrated in Figure 10.

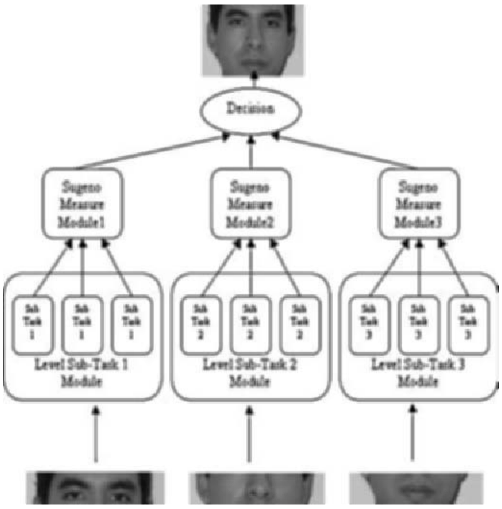


Fig. 9. Final architecture showing inputs and outputs.

4.2 Description of the Integration Module

The integration modules performs its task in two phases. In the first phase, it obtains two matrices. The first matrix, called h, of dimension 3x3, stores the larger index values resulting from the competition for each of the members of the modules. The second matrix , called I, also of dimension 3x3, stores the photograph number corresponding to the particular index.

Once the first phase is finished, the second phase is initiated, in which the decision is obtained. Before making a decision, if there is consensus in the three modules, we can proceed to give the final decision, if there isn't consensus then we have search in matrix g to find the larger index values and then calculate the Sugeno fuzzy measures for each of the modules, using the following formula,

$$g(Mi\ ) = h(A) + h(B) + \lambda\ h(A)\ h(B) \qquad (5)$$

Where λ is equal to 1. Once we have these measures, we select the largest one to show the corresponding photograph.

4.3 Summary of Results

We describe in this section the experimental results obtained with the proposed approach using the 20 photographs as training data. We show in Table 1 the relation between accuracy (measured as the percentage of correct results) and the percentage of noise in the figures.

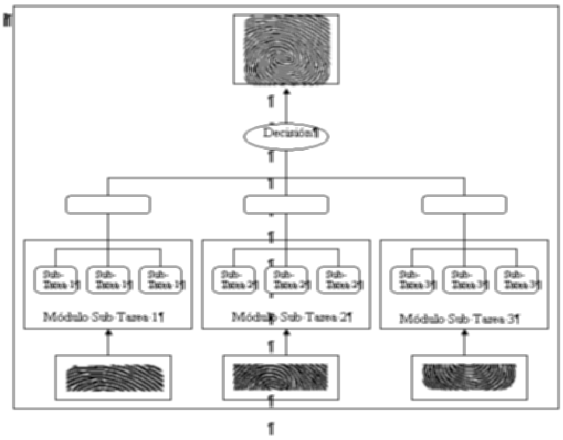


Fig. 10. Final architecture for the fingerprints.

In Table 1 we show the relation that exists between the % of noise that was added in a random fashion to the testing data set, that consisted of the 20 original photographs, plus 200 additional images. We show in Figure 11 sample images with noise.

In Table 2 we show the reliability results for the system. Reliability was calculated as shown in the following formula.

$$\text{Reliability} \quad = \quad \frac{\text{correct results-error}}{\text{correct results}}$$

Table 1. Relation between the % of noise and the % of correct results

% of noise	% accuracy
0	100
10	100
20	100
30	100
40	95
50	100
60	100
70	95
80	100
90	75
100	80

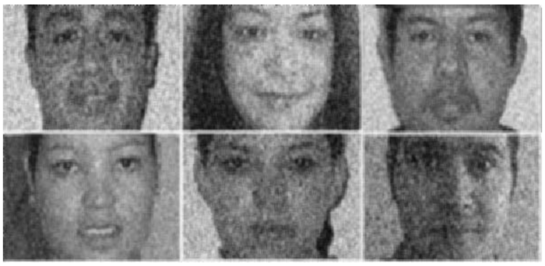


Fig. 11. Sample images with noise.

Table 2. Relation between reliability and accuracy.

% errors	%reliability	%correct results
0	100	100.00
0	100	100.00
0	100	100.00
0	100	100.00
5	94.74	95.00
0	100	100.00
0	100	100.00
5	94.74	95.00
0	100	100.00
25	66.67	75.00
20	75	80.00

We show in Figure 12 a plot relating the percentage of recognition against the number of examples used in the experiments.

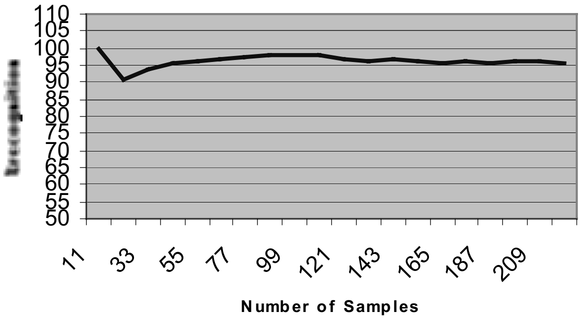


Fig. 12. Relation between recognition and number of examples.

In addition to the results presented before, we also compared the performance of the modular approach, against the performance of a monolithic neural network approach. The conclusion of this comparison was that for this type of input data, the monolithic approach is not feasible, since not only training time is larger, also the recognition is too small for real-world use. We show in Figure 13 a plot showing this comparison but now in a graphical fashion.

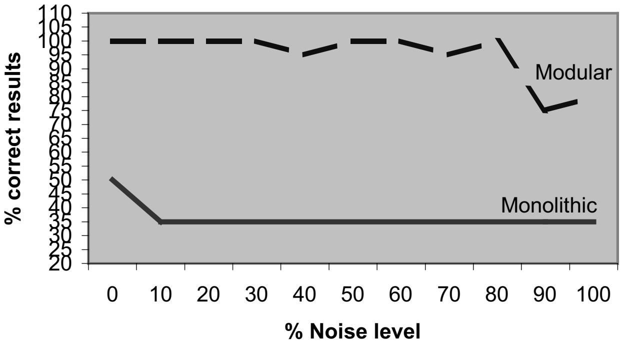


Fig. 13. Comparison between modular and monolithic approach.

5 Conclusions

We showed in this paper the experimental results obtained with the proposed modular approach. In fact, we did achieve a 98.9% recognition rate on the testing data, even with an 80% level of applied noise. For the case of 100% level of applied noise, we did achieve a 96.4 % recognition rate on the testing data. The testing data included 10 photographs for each image in the training data. These 10 photographs were obtained by applying noise in a random fashion, increasing the level of noise from 10 to 100 %, to the training data. We also have to notice that it was achieved a 96.7 % of average reliability with our modular approach. This percentage values was obtained by averaging

In light of the results of our proposed modular approach, we have to notice that using the modular approach for human face pattern recognition is a good alternative with respect to existing methods, in particular, monolithic, gating or voting methods. As future research work, we propose the study of methods for pre-processing the data, like principal components analysis, eigenfaces, or any other method that may improve the performance of the system. Other future work include considering different methods of fuzzy response integration, or considering evolving the number of layers and nodes of the neural network modules.

Acknowlegments

We would like to express our gratitude to the Research Grant Committee of COSNET, for the financial support given to this research work under grant 422.03-P. We would also like to thank CONACYT for the scholarships for the students working in this project (Claudia Gonzalez, Diana Bravo, Felma Gonzalez and Gabriela Martinez).

References

S. Albrecht. (1996), A Modular Neural Network Architecture with Additional Generalization Abilities for High Dimensional Input Vectors.

E.J.W. Boers and H. Kuiper. (1992), “Biological Metaphors and the Design of Modular Artificial Neural Networks”. *Departments of Computer Science and Experimental and Theoretical Psychology at Leid University*, *the Netherlands.*

O. Castillo and P. Melin. (2003), "Soft Computing and Fractal Theory for Intelligent Manufacturing". *Springer-Verlag, Heidelberg, Germany.*

H-C. Fu, Y-P. Lee, C-C Chiang and H-T. Pao. (2001), "Divide-and-Conquer Learning and Modular Perceptron Networks", *IEEE Transaction. Neural Networks,* vol. 12, pp 250-263.

L. K. Hansen and P. Salomon. (1990), "Neural Network Ensembles", *IEEE Transactions on Pattern Analysis and Machine Intelligence. Neural Networks*, vol 12, pp 993-1001.

B. Lu and M. Ito. (1998), "Task Decomposition and module combination based on class relations: modular neural network for pattern classification", *Technical Report, Nagoya Japan.*

R. Murray-Smith and T.A. Johansen. (1997), Multiple Model Approaches to Modeling and Control. *Taylor and Francis.*

A. Quezada (2004), "Reconocimiento de Huellas Digitales Utilizando Redes Neuronales Modulares y Algoritmos Geneticos". *Thesis of Computer Science, Tijuana Institute of Technology.*

E. Ronco and P. Gawthrop. (1995), "Modular neural networks: A State of the Art. Technical Report", *Center for System and Control. University of Glasgow, Glasgow, UK.*

A.J.C. Sharkey. (1998), "Combining Artificial Neural Nets: Ensemble and Modular Multi-Nets Systems", *Ed. Springer-Verlag, New York.*

M. Sugeno. (1974), "Theory of fuzzy integrals and its application", *Doctoral Thesis, Tokyo Institute of Technology.*

R.R. Yager. (1999), "Criteria Aggregations Functions Using Fuzzy Measures and the Choquet Integral", *International Journal of Fuzzy Systems, Vol. 1, No. 2.*

Evolutionary Modeling Using A Wiener Model

Oscar Montiel. CITEDI-IPN. 2498 Roll Dr. 286. San Diego, CA 92154, USA. oross@citedi.mx.

Oscar Castillo. Department of Computer Science, Tijuana Institute of Technology. P.O. Box 4207, Chula Vista CA 91909, USA. ocastillo@tectijuana.mx.

Patricia Melin. Department of Computer Science, Tijuana Institute of Technology. P.O. Box 4207, Chula Vista CA 91909, USA. pmelin@tectijuana.mx.

Roberto Sepúlveda. CITEDI-IPN. Av. del Parque #1310, Mesa de Otay, Tijuana, B. C., México. rsepulve@citedi.mx

Abstract: There exists no standard method for obtaining a nonlinear input-output model using external dynamic approach. In this work, we are using an evolutionary optimization method for estimating the parameters of an NFIR model using the Wiener model structure. Specifically we are using a Breeder Genetic Algorithm (BGA) with fuzzy recombination for performing the optimization work. We selected the BGA since it uses real parameters (it does not require any string coding), which can be manipulated directly by the recombination and mutation operators. For training the system we used amplitude modulated pseudo random binary signal (APRBS). The adaptive system was tested using sinusoidal signals.

Key words: Wiener model, NFIR, NARX, System Identification, Genetic Algorithm, Nonlinear, external dynamics.

1 Introduction

A system is a human conception of a group of independent but interrelated elements comprising a unified whole (Severance, 2001). The key task of system identification (modeling) is to find a best suitable mathematical model between the inputs, outputs and disturbances of a physical system (Ljung, 1999). Nowadays, identifying linear systems has become a routine task and there are available several successful methods for solving the problem in the time or in the frequency domain, using iterative and non-iterative optimization methods for estimating the parameters. Real systems are nonlinear and their properties may change with time, obtaining models for nonlinear systems are more complex than for linear, since any difference in the dynamic behavior of these models can be extremely significant. It is a common practice to treat real systems as linear to some extent, and as a natural consequence to consider a linear model as first choice. If the linear model does not fulfill the expectative, it is necessary to analyze the whole process to search an explanation and solution. A possibility is to change from a linear to a nonlinear model, but sometimes it can be conflicting since if the nonlinear model chosen is not flexible enough its performance can be worst. Nonlinear system identification of a dynamic process is a challenging task and it has received special attention during the past decade (Ikonen, 1999).

In this work, for modeling a nonlinear system we used an input-output model that fits in the class of models known as “external dynamics models”, this name stems from the fact that it can be separated in two well defined parts: an external dynamic filter bank and a nonlinear static approximator (Nelles, 2001). This concept is illustrated in Fig. 1.

There are several nonlinear optimization methods for estimating the parameters of an external dynamic model, most of them perform local searches and they might get trapped in local optima. Global optimization methods can perform global searches for the global optimum, although it is well known the huge computational demand that these methods require. Since nonlinear, local and global, as linear optimization methods have their own advantages and drawbacks, it is a good practice to combine these techniques.

In the field of external dynamic modeling we found some interesting works like: “An iterative method for the identification of nonlinear systems using a Hammerstein model” (Narendra, 1996), here it was proposed the traditional iterative algorithm; in “A new Identification Method for Wiener and Hammerstein Systems” (Guo, 2004), was developed a unified new recursive identification method in the prediction error; in “Worst-Case Identification of Nonlinear Fading Memory Systems” (Dahleh, 1995) was studied the problem of asymptotic identification for fading memory in the presence of bounded noise; in “Identification of Multivariable Hammerstein Systems using Rational Orthonormal Bases” (Gomez, 2004), it is presented a non iterative algorithm for the simultaneous identification of the linear and nonlinear parts of multivariable Hammerstein systems.

For the purpose of putting into context this work, we want to mention two previous works that deal with estimating parameters using evolutionary computation, they are: “The evolutionary learning rule for system identification” (Montiel et al., 2004a) and “Asynchronous hybrid architecture for parametric system identification using fuzzy real coded evolutionary algorithm” (Montiel et al., 2004b). In both papers were shown results of parameter estimation for the Finite Impulse Response (FIR) filter using the BGA.

A different approach is to use genetic programming (GP) mainly focusing on generating algebraic expression for describing a physical system instead of estimating parameters of a model structure. In this branch there are some interesting works, such as, “Finding an Impulse Response Function Using Genetic Programming” (Keane Martin A. et al., 1993) where GP was applied to obtain a symbolic expression for a linear time-invariant system (LTI); two representative works applied to nonlinear systems are: “Multiobejctive Genetic Programming: A Nonlinear System Identification Application” (Rodríguez et al., 1997), and "Identifying Nonlinear Model Structures Using Genetic Programming Techniques" (Winkler S., et al., 2004).

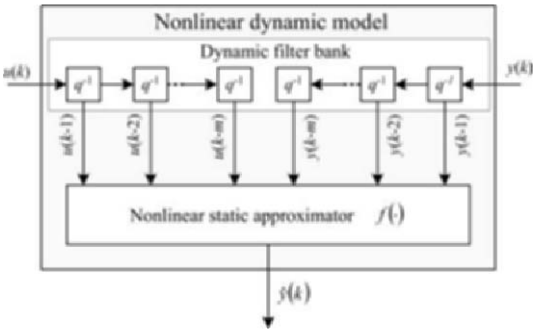


Fig. 1. In the external dynamic approach the model is conceptualized as a dynamic filter bank and a nonlinear static approximator. In principle, it does not matter the model architecture or the static nonlinear approximator.

2 System Description

In Fig. 2 we have a digital signal input $u(k)$ that is fed to the unknown system and to the adaptive system at the same time, in this figure the unknown system is enclosed by dashed lines in a “black box” (Sjoberg, 1995), its output is called the desired response signal and it is represented by $y(k)$. The adaptive system, i.e. the Wiener model will compute a corresponding output signal sample $\hat{y}(k)$ at time k. Both signals, $y(k)$ and $\hat{y}(k)$ are compared subtracting the two samples at time k, to obtain the error signal, $e(k)$,

$$e(k) = y(k) - \hat{y}(k) \tag{1}$$

The adaptive system has the task of representing accurately the signal $y(k)$ at its output, i.e, $y(k) = \hat{y}(k)$. At the unknown system side, we have an additive noisy signal known as the observation noise signal because it corrupts the observation of the signal at the output of the unknown system (Vijay, 1997), then

$$y(k) = y_u(k) + \eta(k) \tag{2}$$

The unknown system can be any system, a simple or a complex system. We used a nonlinear autoregressive with exogenous input (NARX) first order Wiener model given by (Nelles, 2001),

$$y(k) = \arctan\ 0.1 * u(k-1) + 0.9 * \tan(y(k-1))) \tag{3}$$

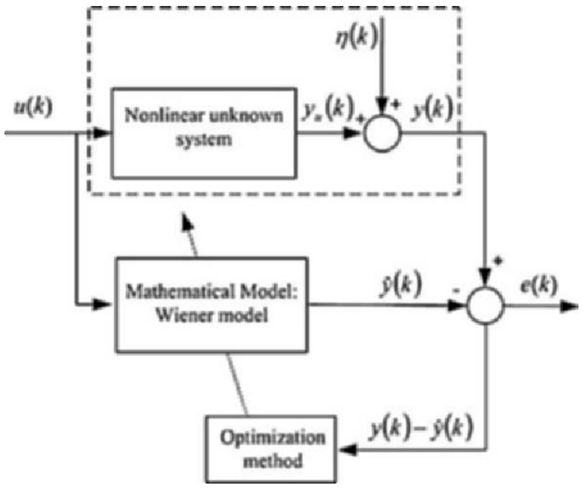


Fig. 2. System identification with noise presence.

The Wiener model structure consists of a linear dynamic block followed by a nonlinear static block. We selected an NFIR Wiener model structure described by (Nelles, 2001)(Gomez Juan C., 2004),

$$\hat{y}(k) = \arctan\left(\sum_{i=0}^{L-1} h_i(k)x(k-i)\right) \tag{4}$$

or in vectorial form

$$\hat{y}(k) = \arctan\left(H^T(k)X(k)\right) \tag{5}$$

where the coefficient vector $H(k)$ is

$$H(k) = [h_0(k)\ \ h_1(k)\ \ \ldots\ \ h_{L-1}(k)] \tag{6}$$

and the input signal vector is given in vectorial form by,

$$X(k) = [x(k)\ \ x(k-1)\ \ \ldots\ \ x(k-L+1)] \tag{7}$$

In real world problems, the identification is successful if we meet with some criterion in the error value. Moreover, in real world problems, the output of the unknown system $y_u(k)$ is contaminated with noise $\eta(k)$. Generally, we do not have direct access to the uncorrupted output $y_u(k)$ of the plant; instead we have a noisy measurement of it. In this case the output is given by equation (2). Then, we can say that the adaptive filter has reached the optimum if we find a value

$y(k) = \hat{y}(k)$ this is achieved when we find an optimum weight vector's parameter *H*, such as

$$H(k) = H_{OPT}(k) \tag{8}$$

3 Evolutionary Optimization Technique

System identification uses a supervised learning method (Jang, 1997), for estimating the optimum parametric vector H(k) it is necessary to use an optimization technique. We selected the evolutionary algorithm known as BGA (Mühlenbein, 1994), and we tested it using the fuzzy recombination (FR) operator (Voigt, 1995). The BGA allows us to represent in a direct way floating point numbers, so the encoding and decoding of each variable is transparent for the user (Deb Kalyanmov, 2002). This algorithm uses a deterministic selection mechanism implemented using truncation selection, only a percent of the best individuals of the whole population is selected for recombining, in this way the survivor of the best individuals is guaranteed, and the extinction of the worst individuals is also guaranteed. The BGA is defined as an eight tuple (Mühlenbein, citeseer.ist.psu.edu/110687.html),

$$BGA = \left(H^0, N, T\%, \Gamma, \Delta, HC, F, term\right) \tag{9}$$

where H^0 is the initial population of size *N, T is* the truncation threshold commonly referred as *T%*, Γ represents the recombination operator, Δ is the mutation operator, *HC* is a hill climbing method (for example: the LS algorithm), *F* is the fitness function and *term* is the termination criterion. In FR for obtaining an offspring it is necessary to recombine the individuals of the population, say $X=(x_1,...,x_n)$, and $Y=(y_1,...,y_n)$ to obtain $Z=(z_1,...,z_n)$ (Voigt, 1995), the offspring z_i is obtained using triangular membership functions, where u_i and y_i are the modes. Equation (10) is the membership function of a normalized triangular fuzzy number, where *m* is the mode and *s* represents the spread of the fuzzy number, for example for the offspring x_i, we have

$$\mu(x_i)_T\{s,m\} = 1 - \frac{2\,|\,m - x_i\,|}{s} \tag{10}$$

and the corresponding probability distribution function (pdf) is

$$\Pr(x_i)_T\{s,m\} = \begin{cases} 0 & x_i < m - s \\ \frac{1}{s^2}\left(x_i + s - m\right) & m - s \le x_i \le m \\ \frac{1}{s^2}\left(-x_i + s + m\right) & m \le x_i \le m + s \\ 0 & x_i > m + s \end{cases} \tag{11}$$

Equation (11) is a unimodal pdf of the allele x_i. For generating the allele successor z_i, we will need a bimodal pdf, so it will be necessary to obtain a unimodal pdf for each allele of each parent. Equation (12) represents the bimodal pdf for the allele z_i,

$$\Pr(z_i)_{BT}\{s_1, m_1, s_2, m_2\} = \frac{1}{2}\left(\Pr(x_i)_T\{s_1, m_1\} + \Pr(y_i)_T\{s_2, m_2\}\right) \tag{12}$$

where the range for each triangular membership function is given in equations (13) and (14)

$$x_i - a \le x_i \le x_i + a \tag{13}$$

$$y_i - a \le y_i \le y_i + a \tag{14}$$

the offspring z_i can lie in one or both of the intervals, the variable a is given by

$$a = e|y_i - x_i|, \qquad e > 0 \tag{15}$$

Using equations (14), (15) and (16) we have that

$$\Pr(z_i) = \Pr(z_i)_{BT}\{e|y_i - x_i|, x_i, e|y_i - x_i|, y_i\} \tag{16}$$

in equation (15), the variable e is the fuzzy spread of the fuzzy numbers, generally e is selected to be 0.5. The mutation operator is applied to each offspring, and the resulting individuals are inserted in the new population $H_r(n)$. The process is repeated until a termination criterion is met.

The goal of the mutation operator Δ is to modify one or more parameters of z_i, the modified objects (i.e., the offsprings) appear in the landscape within a certain distance of the unmodified objects (i.e., the parents). In this way, an offspring Z', where $Z' = (z_1, \ldots, z_n)$ is given by (Mühlenbein Heinz and Schilierkamp-Voosen, 1993)(De Falco et al., 1997),

$$z_i' = z_i \pm range_i \cdot \delta \tag{17}$$

where *range*$_i$ defines the mutation range, and is calculated as (λ·*searchinterval*$_i$). The variable *searchinterval* is defined into the domain of the variable to be mutated, in this case $[-1, 1]$. In the Discrete Mutation operator (DM) λ is normally set to 0.1 or 0.2 and is very efficient in some functions, but also we can set λ to 1. The sign + or – is chosen with probability of 0.5. The variable δ is computed by

$$\delta = \sum_{i=0}^{J-1} \varphi_i 2^{-i} \qquad \varphi_i \in 0,1 \tag{18}$$

Before mutation we set each φ_i equals to 0, then each φ_i is mutated to 1 with probability $p = 1/J$, and only ${}_j$=1 contributes to the sum. On the average there will be just one φ_i with value 1, say φ_j. Then is given by:

$$\delta = 2^{-i} \tag{19}$$

In formula (18), J is a parameter originally related to the machine precision, i.e. the number of bits used to represent a real variable in the machine we are working with, traditionally J used values of 8 and 16. In practice, however, the value of J is related to the expected value of mutation steps, in other words, the higher J is, the more fine-grained is the resultant mutation operator (De Falco, 1997).

4 Training Signal Generation

For identifying a process adequately is necessary to use an appropriated excitation signal. Nonlinear processes require excitation of their dynamic and static properties in all relevant points, so it will be necessary to use a sequence that combines excitation of both parts, static and dynamic. For these reasons it is common to use an amplitude modulated pseudo random binary signal (APRBS). Fig. 3 shows an APRBS signal (solid line), it is the input to both systems, and it is used for training the adaptive model, in this figure is magnified the minimal hold time (T_h). This time is the minimal step size of the signal, i.e., it is the shortest period of time for which the signal stays constant. For a given signal length, T_h will have a direct influence in the number of steps and the frequency characteristics. The selection of T_h is different for linear and nonlinear SI, in linear SI this time is selected equal to the sampling time, at the other hand for nonlinear SI T_h should be selected neither too small nor too large. If we select T_h too small then the system will not have time to settle, only operating conditions around $y_0 \approx (u_{\max} + u_{\min})/2$ will be covered (Nelles, 2001). If T_h is too large, then for a given signal only a reduced set of operating conditions will be covered, the amount of points mainly depends on the number of steps with different amplitude. As a consequence, if we do not modulate in amplitude the signal and we only use pseudo random signals (PRBS), we will have only two operating conditions, one for each signal's value. In Fig. 3 also we have a dashed line which is the output of the unknown system. Fig. 4 shows the data distribution of the training data of Fig. 3, there are some holes and they are random located, depending on the amplitude levels. Nevertheless, the holes trend to disappear or at least they will become smaller as the length of the training signal increases.

5 Experimental Results

In Fig. 5, we are showing an outline sketch of the software implementation that we used for identifying the unknown system represented by equation (3). This software uses the evolutionary floating point algorithm known as BGA explained in section 3. We used a population size of 1,000 individuals floating point coded.

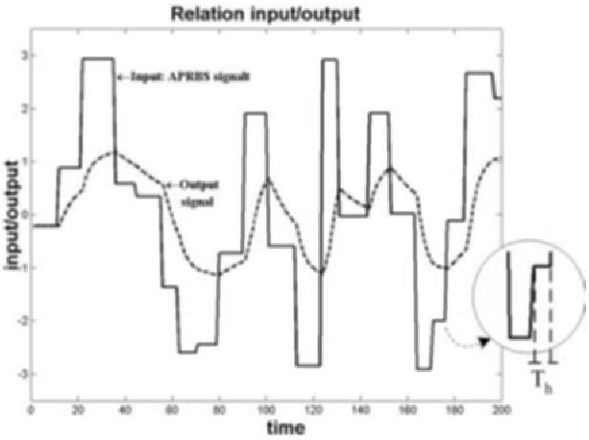


Fig. 3. At the input of both systems (unknown and adaptive model structure) we applied an APRBS signal (solid). This APRBS signal consists of 200 samples, its amplitude is in the interval of [-3,3]. The minimal hold time T_h is 5 samples and the maximum hold time Tm_h is 15 samples. Here, we are showing the unknown system's output with a dashed line.

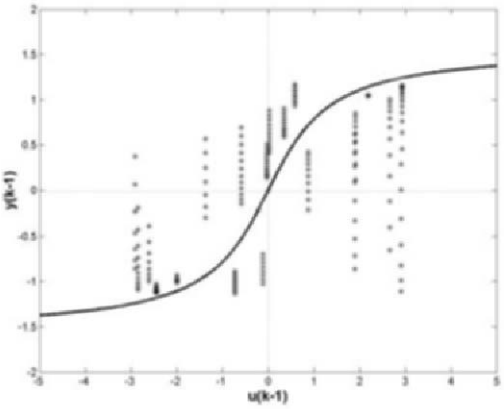
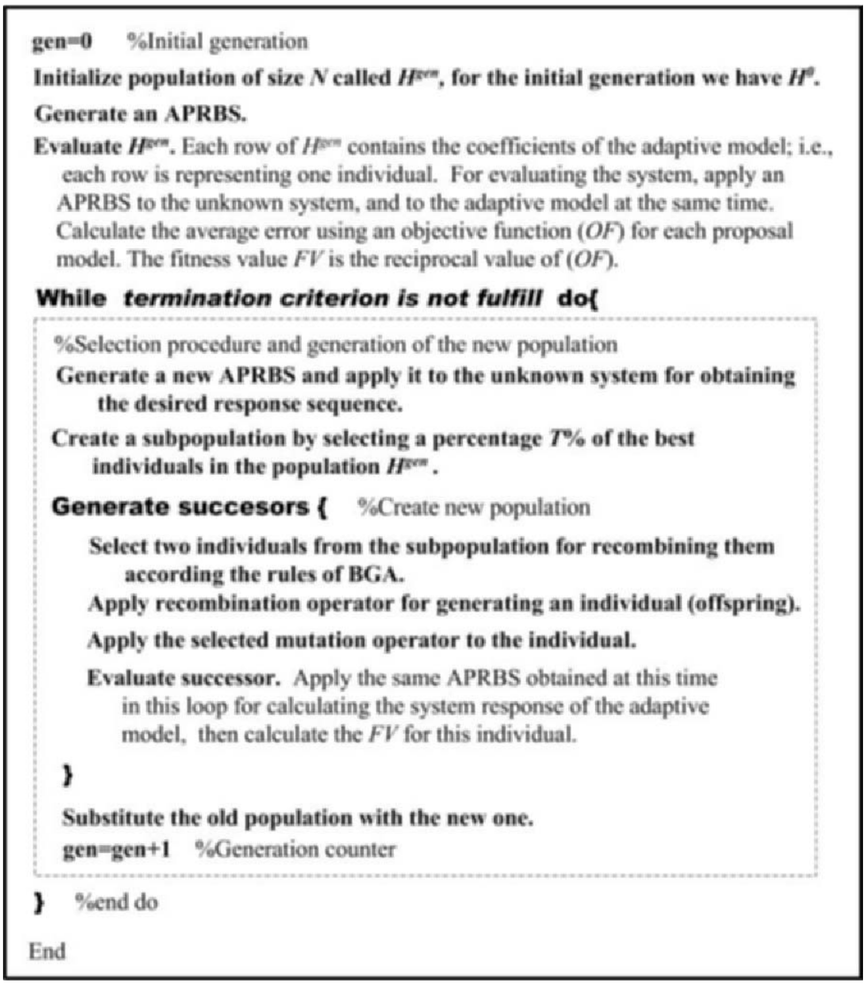


Fig. 4. Data distribution of the training signal shown in Fig. 3.

We implemented the fuzzy recombination (FR) operator described by equations 10 to 16, where we used for the variable *e* a value of 0.5.

For the mutation operator, we used the discrete mutation operator (DM) with a λ value of 0.1 and a range value of [-1,1] for the variable *searchinterval*. We calculated equation 18 using *J*=16. The new mutated offspring was calculated using equation 17, using the specifications mentioned above.

The BGA uses a deterministic selection mechanism in the sense that only the best individuals are selected for creating the offspring by means of applying the recombination and mutation operators, moreover, we saved the best individual, i.e. the individual with the highest fitness value through generations, this is shown in Fig. 6. with small circles ‘o’. The average fitness value of the whole population at each generation is shown in Fig. 6 with ‘x’.



```
gen=0   %Initial generation
Initialize population of size $N$ called $H^{gen}$, for the initial generation we have $H^0$.
Generate an APRBS.
Evaluate $H^{gen}$. Each row of $H^{gen}$ contains the coefficients of the adaptive model; i.e.,
    each row is representing one individual. For evaluating the system, apply an
    APRBS to the unknown system, and to the adaptive model at the same time.
    Calculate the average error using an objective function (OF) for each proposal
    model. The fitness value FV is the reciprocal value of (OF).
While termination criterion is not fulfill do{
    %Selection procedure and generation of the new population
    Generate a new APRBS and apply it to the unknown system for obtaining
        the desired response sequence.
    Create a subpopulation by selecting a percentage T% of the best
        individuals in the population $H^{gen}$ .
    Generate succesors {   %Create new population
        Select two individuals from the subpopulation for recombining them
            according the rules of BGA.
        Apply recombination operator for generating an individual (offspring).
        Apply the selected mutation operator to the individual.
        Evaluate successor. Apply the same APRBS obtained at this time
            in this loop for calculating the system response of the adaptive
            model, then calculate the FV for this individual.
    }
    Substitute the old population with the new one.
    gen=gen+1   %Generation counter
}   %end do
End
```

Fig. 5. Outline of the software implementation for solving the nonlinear system identification problem. We tested the software for obtaining models of the NFIR type model structure. This software can be applied for solving the generic nonlinear system identification problem known as nonlinear autoregressive moving average with exogenous input (NARMAX) type.

At the adaptive side, we used a Wiener model structure for modeling the system, the parameters were estimated using the BGA with the above mentioned characteristics. For training the adaptive system, we made in Fig. 2, $\eta(k)=0$; but for testing the system we used random noise with normal distribution with mean equals to zero and variance $\sigma^2 = 0.1$. We ran the algorithm 400 generations, at each generation we applied an APRBS to the systems: the unknown, and the adaptive model. This signal (APRBS) was generated with the next characteristics: 200 samples, with random amplitude in the interval of [-3,3], the minimal hold time T_h is 5 samples, moreover, we used maximum hold time Tm_h consisting of 15 samples.

For testing the system we applied to the unknown system a sinusoidal signal, this signal and its corresponding output is shown in Fig. 7. In Fig. 8 we have the data distribution of this signal at the unknown system side.

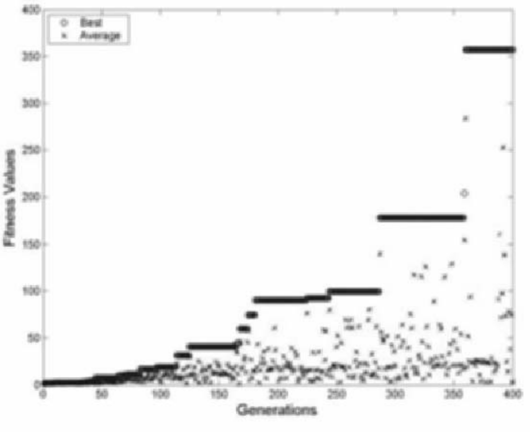


Fig. 6. Here we are showing the best fitness value at each generation, we preserved the best individual at each generation. At generation 360 we found the best fitness values of 357.1027. The average square error is 0.0028, we used 80 samples for computing this value. The fittest individual at each generation is plotted using a small 'o', and the average fitness value of the population at each generation is shown with 'x'. Note that although the selection procedure is deterministic and we are including the fittest individual through generation, the algorithm still have good explorative characteristics, preventing the population to fully converge.

6 Conclusions

Global search capabilities of evolutionary algorithms can be exploited for estimating parameters in nonlinear input-output models structures such as in a Wiener model structure. This is an off-line option for system identification since it is highly computational time demanding, but it could be implemented in on-line application working in a second plane, searching for better models. Using an NFIR model structure gives us stability since it does not have feedback, but the price to pay is that we will need higher dynamic order for describing the process dynamics properly. The NFIR model can represent an unstable model for the first L samples, where L is the filter's order. For improving the performance of this method it is necessary to implement faster evolutionary strategies and better fitness functions.

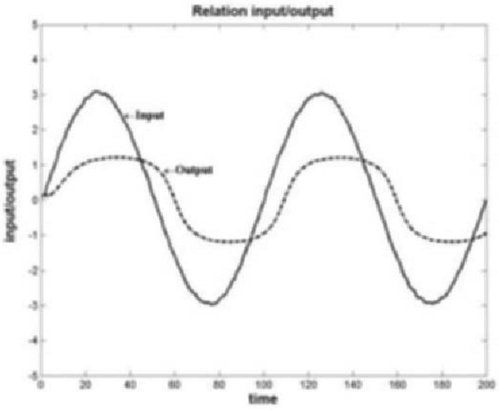


Fig. 7. Validation data. We used a sinusoidal sequence for testing the system (solid line), the system's output is shown with a dashed line.

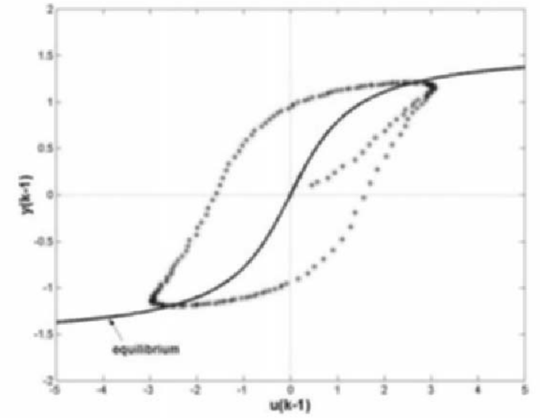


Fig. 8. Unknown system data distribution. for the sinusoidal signal of Fig. 7.

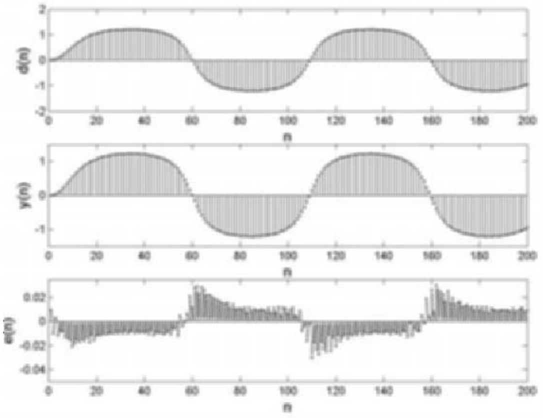


Fig. 9. In this graphic we have three plots; they were obtained using the best parametric vector found for the NFIR model structure. The upper one is the unknown system output; at the middle, we have the NFIR model output; and at the bottom we have the error sequence obtained subtracting the two preceding plots. For the error signal we found that the maximum value of this sequence is 0.02996, the mean value is 0.0002821, the median value is -0.003728, and the standard deviation value is 0.01324. These values were obtained considering 200 samples.

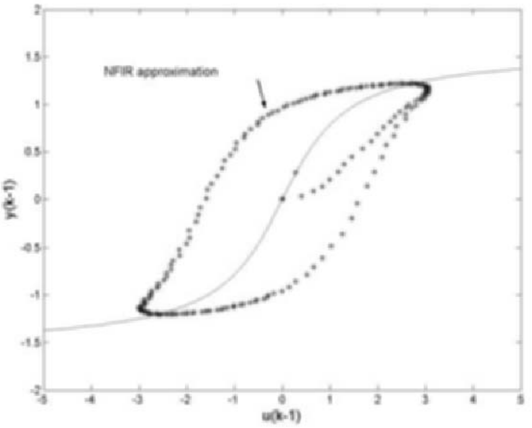


Fig. 10. Data distribution at the adaptive system side. This signal was obtained when we applied a sinusoidal excitation (Fig. 7) at the NFIR model once it was trained; i.e., with the optimal parameters.

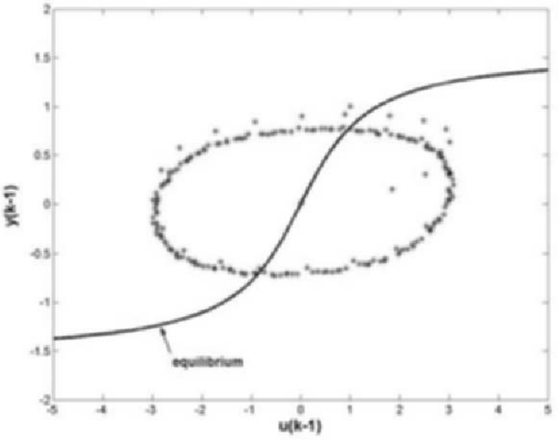


Fig. 11. This figure was obtained when we increase the sinusoidal frequency almost two times.

7 Acknowledgment

The authors thank “Comisión de Operación y Fomento de Actividades Académicas del I.P.N.”, and “Instituto Tecnológico de Tijuana” for supporting our research activities.

8 References

Dahleh M. A., E. D. Sontag, D. N. C. Tse, and J. N. Tsitsiklis, 1995, “Worst-case identification of nonlinear fading memory systems”, Automatica, vol. 31, pp. 303—308. http://citeseer.ist.psu.edu/dahleh95worstcase.html

De Falco, I., Della Cioppa, A., Natale, P., Tarantino, E. (1997), “Artificial Neural Networks Optimization by means of Evolutionary Algorithms”, http://citeseer.nj.nec.com/defalco97artificial.html

Deb Kalyanmoy (2002), “Multi-Objective Optimization using Evolutionary Algorithms”, John Wiley & Sons, LTD, New York, USA.

Gomez Juan and Baeyens Enrique (2004), **“**Identification of Multivariable Hammerstein Systems using Rational Orthonormal Bases”, http://citeseer.ist.psu.edu/421047.html.

Gómez Juan C., Enrique Baeyens (2004) , “Identification of Nonlinear Systems using Orthonormal Bases”, http://citeseer.ist.psu.edu/596443.html

Guo Fen (2004), “A New Identification Method for Wiener and Hammerstein Systems”, Institut für Angewandte Informatik, http://bibliothek.fzk.de/zb/berichte/FZKA6955.pdf

Ikonen E., Najim K. (1999), “Learning control and modelling of complex industrial processes, Overview report of our activities within the European Science Foundation's programme on Control of Complex Systems (COSY) Theme 3: Learning control". http://cc.oulu.fi/~iko/lccs.html

Keane Martin A., Koza John R., Rice James P. (1993). Finding an impulse response function using genetic programming. In Proceedings of the 1993 American Control Conference, volume 3, pages 2345--2350, San Francisco, CA, 2.-4. June 1993. IEEE, New York. http://citeseer.ist.psu.edu/keane93finding.html

Ljung Lennart (1999), "System Identification. Theory for the User. Second Edition", Prentice Hall PTR, USA.

Nelles Oliver (2001), "Nonlinear System Identification. From Classical Approaches to Neural networks and Fuzzy Models", Springer-Verlag Berlin Heidelberg. Germany. 2001. pp. 15, 457-511.

Montiel R. Oscar, Oscar Castillo, Roberto Sepúlveda, Patricia Melin (2004a) , "The evolutionary learning rule for system identification", Applied Soft Computing Journal. Special issue: Soft Computing for Control of Non-Linear Dynamical Systems. Volume 3, Issue 4. December 2003. pp. 343-352

Montiel Oscar, Oscar Castillo, Patricia Melin, Roberto Sepúlveda (2004b) , "Asynchronous hybrid architecture for parametric system identification using fuzzy real coded evolutionary algorithm", Nonlinear Studies, Volume 11, Number 1.

Mühlenbein Heinz, Dirk Schlierkamp-Voosen (1994) , "The science of breeding and its application to the breeder genetic algorithm BGA". Evolutionary Computation, 1(4):335-360.

Mühlenbein Heinz, Evolutionary Algorithms: Theory and Applications, http://citeseer.ist.psu.edu/110687.html.

Mühlenbein Heinz and Schilierkamp-Voosen (1993) , "Predictive Model for Breeder Genetic Algorithm", Evolutionary Computation. 1(1): 25-49.

Narendra K. S. and P. G. Gallman (1996), "An iterative method for the identification of nonlinear systems using a Hammerstein model". IEEE Transactions on Automatic Control, AC-11:546-550. July 1966.

Rodríguez Katya Vázquez, Fonseca Carlos M. Fleming Peter J. (1997) . Multiobjective Genetic Programming: A Nonlinear System Identification Application. Late Breaking Papers at the Genetic Programming 1997 Conference, Editor John R. Koza, Standford Bookstore, USA, pp. 207-212.

Severance Frank L. (2001), "System Modeling and Simulation. An Introduction", John Wiley & Sons Ltd., UK.

Sjoberg, J., Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.-Y. Glorennec, H. Hjalmarsson and A. Juditsky (1995). "Nonlinear black-box modeling in system identification: a unified overview". Automatica 31(12), 1691--1724. http://citeseer.nj.nec.com/sjoberg95nonlinear.html

Vijay K. Madisetti, Douglas B. Williams (1997). "The Digital Signal Processing Handbook", A CRC Handbook Published in Cooperation with IEEE Press, pp. 15-1, 18-1, 18-12, 20-1, 20-4.

Jang J.-S.R., C.-T. Sung, E. Mizutani (1997), "Neuro-Fuzzy and Soft Computing. A Computational Approach to Learning and Machine Intelligence". Prentice Hall. NJ, USA.

Voigt H.M., Mühlenbein, D. Cvetkovic (1995) , "Fuzzy Recombination for the Breeder Genetic Algorithm", Proceedings of the Sixth International Conference on Genetic Algorithms, published by Morgan Kaufmann, pp. 1104-111.

Winkler S., Affenzeller M., Wagner S. (2004), "Identifying Nonlinear Model Structures Using Genetic Programming Techniques". *Cybernetics and Systems 2004*, pp. 689-694. Austrian Society for Cybernetic Studies, 2004.

Evolutionary Computing for Topology Optimization of Fuzzy Systems in Intelligent Control

Oscar Castillo, Gabriel Huesca, and Fevrier Valdez
Department of Computer Science
Tijuana Institute of Technology
Tijuana, Mexico

Abstract. We describe in this paper the use of hierarchical genetic algorithms for fuzzy system optimization in intelligent control. In particular, we consider the problem of optimizing the number of rules and membership functions using an evolutionary approach. The hierarchical genetic algorithm enables the optimization of the fuzzy system design for a particular application. We illustrate the approach with the case of intelligent control in a medical application. Simulation results for this application show that we are able to find an optimal set of rules and membership functions for the fuzzy system.

Keywords: Evolutionary Computing, Intelligent Control, Fuzzy Logic, Fuzzy Control

1 Introduction

We describe in this paper the application of a Hierarchical Genetic Algorithm (HGA) for fuzzy system optimization (Man et al. 1999). In particular, we consider the problem of finding the optimal set of rules and membership functions for a specific application (Yen and Langari 1999). The HGA is used to search for this optimal set of rules and membership functions, according to the data about the problem. We consider, as an illustration, the case of a fuzzy system for intelligent control.

Fuzzy systems are capable of handling complex, non-linear and sometimes mathematically intangible dynamic systems using simple solutions (Jang et al. 1997). Very often, fuzzy systems may provide a better performance than conventional non-fuzzy approaches with less development cost (Procyk and Mamdani 1979). However, to obtain an optimal set of fuzzy membership functions and rules is not an easy task. It requires time, experience and skills of the designer for the tedious fuzzy tuning exercise. In principle, there is no general rule or method for the fuzzy logic set-up, although a heuristic and iterative procedure for modifying the membership functions to improve performance has been proposed. Recently, many researchers have considered a number of intelligent schemes for the task of tuning the fuzzy system. The noticeable Neural Network (NN) approach (Jang and Sun 1995) and the Genetic Algorithm (GA) approach (Homaifar and McCormick 1995) to optimize either the membership functions or rules, have become a trend for fuzzy logic system development.

The HGA approach differs from the other techniques in that it has the ability to reach an optimal set of membership functions and rules without a known fuzzy system topology (Tang et al. 1998). During the optimization phase, the membership functions need not be fixed. Throughout the genetic operations (Holland 1975), a reduced fuzzy system including the number of membership functions and fuzzy rules will be generated (Yoshikawa et al. 1996). The HGA approach has a number of advantages:

1) An optimal and the least number of membership functions and rules are obtained
2) No pre-fixed fuzzy structure is necessary, and
3) Simpler implementing procedures and less cost are involved.

We consider in this paper the case of automatic anesthesia control in human patients for testing the optimized fuzzy controller. We did have, as a reference, the best fuzzy controller that was developed for the automatic anesthesia control (Karr and Gentry 1993, Lozano 2003), and we consider the optimization of this controller using the HGA approach. After applying the genetic algorithm the number of fuzzy rules was reduced from 12 to 9 with a similar performance of the fuzzy controller. Of course, the parameters of the membership functions were also tuned by the genetic algorithm. We did compare the simulation results of the optimized fuzzy controllers obtained with the HGA against the best fuzzy controller that was obtained previously with expert knowledge, and control is achieved in a similar fashion. Since simulation results are similar, and the number of fuzzy rules was reduced, we can conclude that the HGA approach is a good alternative for designing fuzzy systems. We have to mention that Type-2 fuzzy systems are considered in this research work, which are more difficult to design and optimize.

2 Genetic Algorithms for Optimization

In this paper, we used a floating-point genetic algorithm (Castillo and Melin 2001) to adjust the parameter vector θ, specifically we used the Breeder Genetic Algorithm (BGA). The genetic algorithm is used to optimize the fuzzy system for control that will be described later (Castillo and Melin 2003). A BGA can be described by the following equation:

$$\mathrm{BGA}=(\mathrm{P}_g^0, \mathrm{N}, \mathrm{T}, \Gamma,\Delta, \mathrm{HC}, \mathrm{F}, \mathrm{term}) \qquad (1)$$

where: P_g^0=initial population, N=the size of the population, T=the truncation threshold, Γ=the recombination operator, Δ=the mutation operator, HC=the hill climbing method, F=the fitness function, *term*=the termination criterion.

The BGA uses a selection scheme called truncation selection. The %T best individuals are selected and mated randomly until the number of offspring is equal the size of the population. The offspring generation is equal to the size of the population. The offspring generation replaces the parent population. The best individual found so far will remain in the population. Self-mating is prohibited (Melin and Castillo 2002). As a recombination operator we used "extended intermediate recombination", defined as: If $x = (x_i, \ldots x_n)$ *and* y $y=(y_1,\ldots,y_n)$ are the parents, then the successor $z=(z_1,\ldots,z_n)$ is calculated by:

$$z_i = x_i + \alpha_i(y_i - x_i) \qquad i = 1,\ldots n \qquad (2)$$

The mutation operator is defined as follows: A variable x_i is selected with probability p_m for mutation. The BGA normally uses $p_m = 1/n$. At least one variable will be mutated. A value out of the interval $[-range_i, range_i]$ is added to the variable. $range_i$ defines the mutation range. It is normally set to $(0.1\ x\ searchinterval_i)$. searchinterval$_i$ is the domain of definition for variable x_i. The new value z_i is computed according to

$$z_i = x_i \pm range_i \cdot \delta \qquad (3)$$

The + or – sign is chosen with probability 0.5. δ is computed from a distribution which prefers small values. This is realized as follows

$$\delta = \sum_{i=0}^{15} \alpha_i 2^i \qquad \alpha_i \in 0{,}1 \qquad (4)$$

Before mutation we set $\alpha_i=0$. Then each α_i is mutated to 1 with probability $p_\delta=1/16$. Only $\alpha_i=1$ contributes to the sum. On the average there will be just one α_i with value 1, say α_j. Then δ is given by

$$\delta = 2^{-j} \qquad (5)$$

The standard BGA mutation operator is able to generate any point in the hypercube with center x defined by $x_i \pm range_i$. But it generates values much more often in the neighborhood of x. In the above standard setting, the mutation operator is able to locate the optimal x_i up to a precision of $ramge_i \cdot 2^{-150}$.

To monitor the convergence rate of the LMS algorithm, we computed a short term average of the squared error $e^2(n)$ using

$$ASE(m) = \frac{1}{K} \sum_{k=n+1}^{n+K} e^2(k) \qquad (6)$$

where $m=n/K=1,2,\ldots.$ The averaging interval K may be selected to be (approximately) K=10N. The effect of the choice of the step size parameter Δ on the convergence rate of LMS algorithm may be observed by monitoring the ASE(m).

2.1 Genetic Algorithm for Optimization

The proposed genetic algorithm is as follows:

1. We use real numbers as a genetic representation of the problem.
2. We initialize variable i with zero (i=0).
3. We create an initial random population P_i, in this case (P_0). Each individual of the population has n dimensions and, each coefficient of the fuzzy system corresponds to one dimension.

4. We calculate the normalized fitness of each individual of the population using linear scaling with displacement (Melin and Castillo 2002), in the following form:

$$f_i^{'} = f_i + \frac{1}{N}\sum \left|f_i\right| + \left|\min_i(f_i)\right| \qquad \forall i$$

5. We normalize the fitness of each individual using:

$$F_i = \frac{f_i^{'}}{\sum_{i=1}^{N} f_i^{'}} \qquad \forall\, i$$

6. We sort the individuals from greater to lower fitness.
7. We use the truncated selection method, selecting the %T best individuals, for example if there are 500 individuals and, then we select 0.30*500=150 individuals.
8. We apply random crossover, to the individuals in the population (the 150 best ones) with the goal of creating a new population (of 500 individuals). Crossover with it self is not allowed, and all the individuals have to participate. To perform this operation we apply the genetic operator of extended intermediate recombination as follows:

If $x=(x_1,...,x_n)$ and $y=(y_1,...,y_n)$ are the parents, then the successors $z=(z_1,...,z_n)$ are calculated by, $z_i=x_i+\alpha_i(y_i-x_i)$ *for* $i=1,...,n$ where α is a scaling factor selected randomly in the interval $[-d,1+d]$. In intermediate recombination $d=0$, and for extended $d>0$, a good choice is $d=0.25$, which is the one that we used.

9. We apply the mutation genetic operator of BGA. In this case, we select an individual with probability $p_m=1/n$ (where n represents the working dimension, in this case n=25, which is the number of coefficients in the membership functions). The mutation operator calculates the new individuals z_i of the population in the following form: $z_i=x_i\pm range_i\ \delta$ we can note from this equation that we are actually adding to the original individual a value in the interval: $[-range_i,range_i]$ the range is defined as the search interval, which in this case is the domain of variable x_i, the sign ± is selected randomly with probability of 0.5, and is calculated using the following formula,

$$\delta = \sum_{i=0}^{m-1} \alpha_i 2^{-i} \qquad \alpha_i \in 0,1$$

Common used values in this equation are $m=16$ y $m=20$. Before mutation we initiate with $\alpha_i=0$, then for each α_i we mutate to 1 with probability $p_\delta=1/m$.

10. Let i=i+1, and continue with step 4.

3. Evolution of Fuzzy Systems

Ever since the very first introduction of the fundamental concept of fuzzy logic by Zadeh in 1973, its use in engineering disciplines has been widely studied. Its main attraction undoubtedly lies in the unique characteristics that fuzzy logic systems possess. They are capable of handling complex, non-linear dynamic systems using simple solutions. Very often, fuzzy systems provide a better performance than conventional non-fuzzy approaches with less development cost.

However, to obtain an optimal set of fuzzy membership functions and rules is not an easy task. It requires time, experience, and skills of the operator for the tedious fuzzy tuning exercise. In principle, there is no general rule or method for the fuzzy logic set-up. Recently, many researchers have considered a number of intelligent techniques for the task of tuning the fuzzy set. Here, another innovative scheme is described (Tang et al. 1998). This approach has the ability to reach an optimal set of membership functions and rules without a known overall fuzzy set topology. The conceptual idea of this approach is to have an automatic and intelligent scheme to tune the membership functions and rules, in which the conventional closed loop fuzzy control strategy remains unchanged, as indicated in Figure 1.

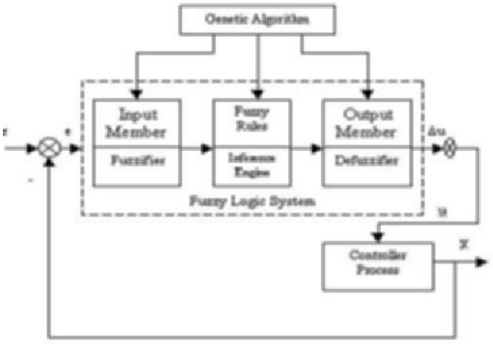


Figure 1 Genetic algorithm for a fuzzy control system.

In this case, the chromosome of a particular system is shown in Figure 2. The chromosome consists of two types of genes, the control genes and parameter genes. The control genes, in the form of bits, determine the membership function activation, whereas the parameter genes are in the form of real numbers to represent the membership functions.

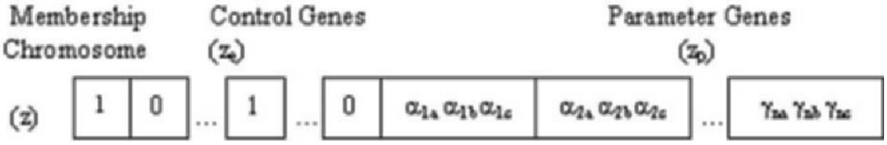


Figure 2 Chromosome structure for the fuzzy system.

To obtain a complete design for the fuzzy control system, an appropriate set of fuzzy rules is required to ensure system performance. At this point it should be stressed that the introduction of the control genes is done to govern the number of fuzzy subsets in the system. Once the formulation of the chromosome has been set for the fuzzy membership functions and rules, the genetic operation cycle can be performed. This cycle of operation for the fuzzy control system optimization using a genetic algorithm is illustrated in Figure 3. There are two population pools, one for storing the membership chromosomes and the other for storing the fuzzy rule chromosomes. We can see this in Figure 3 as the membership population and fuzzy rule population, respectively. Considering that there are various types of gene structure, a number of different genetic operations can be used. For the crossover operation, a one-point crossover is applied separately for both the control and parameter genes of the membership chromosomes within certain operation rates. There is no crossover operation for fuzzy rule chromosomes since only one suitable rule set can be assisted.

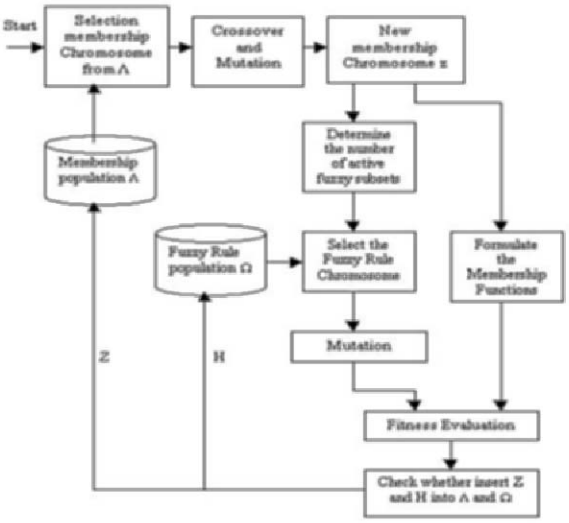

Start
Selection membership Chromosome from Λ
Crossover and Mutation
New membership Chromosome z
Membership population Λ
Determine the number of active fuzzy subsets
Fuzzy Rule population Ω
Select the Fuzzy Rule Chromosome
Formulate the Membership Functions
Mutation
Z
H
Fitness Evaluation
Check whether insert Z and H into Λ and Ω

Figure 3 Genetic cycle for fuzzy system optimization.

Bit mutation is applied for the control genes of the membership chromosome. Each bit of the control gene is flipped if a probability test is satisfied (a randomly generated number is smaller than a predefined rate). As for the parameter genes, which are real number represented, random mutation is applied.

The fitness function can be defined in this case as follows:

$$f_i = \Sigma / y\,(k) - r\,(k) / \qquad (7)$$

where Σ indicates the sum for all the data points in the training set, and y(k) represents the real output of the fuzzy system and r(k) is the reference output. This fitness value measures how well the fuzzy system is approximating the real data of the problem.

4 Type-2 Fuzzy Logic

The concept of a type-2 fuzzy set, was introduced by Zadeh (Melin and Castillo 2002) as an extension of the concept of an ordinary fuzzy set (henceforth called a "type-1 fuzzy set"). A type-2 fuzzy set is characterized by a fuzzy membership function, i.e., the membership grade for each element of this set is a fuzzy set in [0,1], unlike a type-1 set (Castillo and Melin 2001, Melin and Castillo 2002) where the membership grade is a crisp number in [0,1]. Such sets can be used in situations where there is uncertainty about the membership grades themselves, e.g., an uncertainty in the shape of the membership function or in some of its parameters. Consider the transition from ordinary sets to fuzzy sets (Castillo and Melin 2001). When we cannot determine the membership of an element in a set as 0 or 1, we use fuzzy sets of type-1. Similarly, when the situation is so fuzzy that we have trouble determining the membership grade even as a crisp number in [0,1], we use fuzzy sets of type-2.

Example: Consider the case of a fuzzy set characterized by a Gaussian membership function with mean m and a standard deviation that can take values in $[\sigma_1,\sigma_2]$, i.e.,

$$\mu(x)=\exp\{-\tfrac{1}{2}[(x-m)/\sigma]^2\};\quad \sigma\in[\sigma_1,\sigma_2] \qquad (8)$$

Corresponding to each value of σ, we will get a different membership curve (Figure 4). So, the membership grade of any particular x (except x=m) can take any of a number of possible values depending upon the value of σ, i.e., the membership grade is not a crisp number, it is a fuzzy set. Figure 4 shows the domain of the fuzzy set associated with x=0.7.

The basics of fuzzy logic do not change from type-1 to type-2 fuzzy sets, and in general, will not change for any type-n (Castillo and Melin 2003). A higher-type number just indicates a higher "degree of fuzziness". Since a higher type changes the nature of the membership functions, the operations that depend on the membership functions change; however, the basic principles of fuzzy logic are independent of the nature of membership functions and hence, do not change. In Figure 5 we show the general structure of a type-2 fuzzy system. We assume that both antecedent and consequent sets are type-2; however, this need not necessarily be the case in practice.

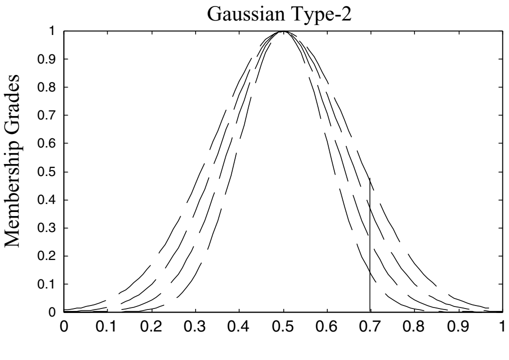


Figure 4 A type-2 fuzzy set representing a type-1 set with uncertain deviation.

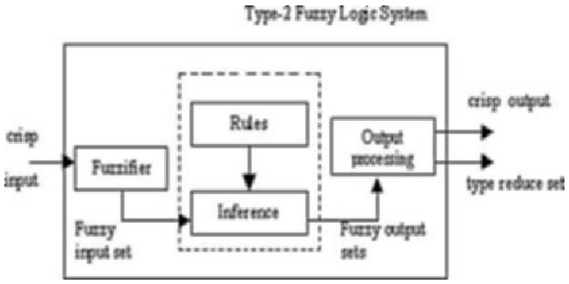


Figure 5. Structure of a type-2 fuzzy system.

The structure of the type-2 fuzzy rules is the same as for the type-1 case because the distinction between type-2 and type-1 is associated with the nature of the membership functions. Hence, the only difference is that now some or all the sets involved in the rules are of type-2. In a type-1 fuzzy system, where the output sets are type-1 fuzzy sets, we perform defuzzification in order to get a number, which is in some sense a crisp (type-0) representative of the combined output sets. In the type-2 case, the output sets are type-2; so we have to use extended versions of type-1 defuzzification methods. Since type-1 defuzzification gives a crisp number at the output of the fuzzy system, the extended defuzzification operation in the type-2 case gives a type-1 fuzzy set at the output. Since this operation takes us from the type-2 output sets of the fuzzy system to a type-1 set, we can call this operation "type reduction" and call the type-1 fuzzy set so obtained a "type-reduced set". The type-reduced fuzzy set may then be defuzzified to obtain a single crisp number; however, in many applications, the type-reduced set may be more important than a single crisp number. Type-2 sets can be used to convey the uncertainties in membership functions of type-1 fuzzy sets, due to the dependence of the membership functions on available linguistic and numerical information.

5 Application to Intelligent Control

We consider the case of controlling the anesthesia given to a patient as the problem for finding the optimal fuzzy system for control (Lozano 2003). The complete implementation was done in the MATLAB programming language. The fuzzy systems were build automatically by using the Fuzzy Logic Toolbox, and genetic algorithm was coded directly in the MATLAB language. The fuzzy systems for control are the individuals used in the genetic algorithm, and these are evaluated by comparing them to the ideal control given by the experts. In other words, we compare the performance of the fuzzy systems that are generated by the genetic algorithm, against the ideal control system given by the experts in this application.

5.1 Anesthesia Control Using Fuzzy Logic

The main task of the anesthesist, during and operation, is to control anesthesia concentration. In any case, anesthesia concentration can't be measured directly. For this reason, the anesthesist uses indirect information, like the heartbeat, pressure, and motor activity. The anesthesia concentration is controlled using a medicine, which can be given by a shot or by a mix of gases. We consider here the use of isoflurance, which is usually given in a concentration of 0 to 2% with oxygen. In Figure 6 we show a block diagram of the controller.

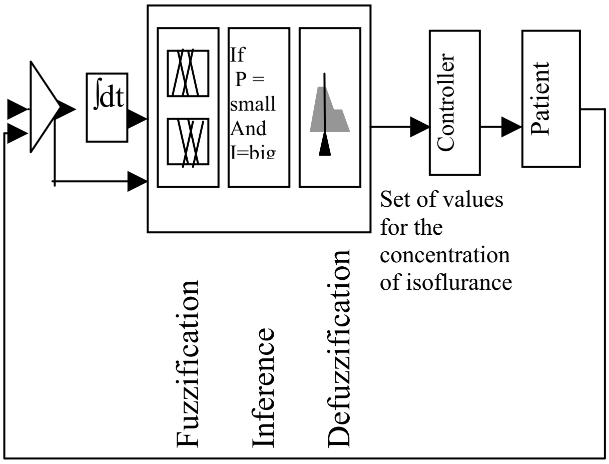


Figure 6 Architecture of the fuzzy control system.

The air that is exhaled by the patient contains a specific concentration of isoflurance, and it is re-circulated to the patient. As consequence, we can measure isoflurance concentration on the inhaled and exhaled air by the patient, to estimate isoflurance concentration on the patient's blood. From the control engineering point of view, the task by the anesthesist is to maintain anesthesia concentration between the high level W (threshold to wake up) and the low level E (threshold to success). These levels are difficult to be determine in a changing environment and also are dependent on the patient's condition. For this reason, it is important to automate this anesthesia control, to perform this task more efficiently and accurately, and also to free the anesthesist from this time consuming job. The anesthesist can then concentrate in doing other task during operation of a patient.

The first automated system for anesthesia control was developed using a PID controller in the 60's. However, this system was not very succesful due to the non-linear nature of the problem of anesthesia control. After this first attempt, adaptive control was proposed to automate anesthesia control, but robustness was the problem in this case. For these reasons, fuzzy logic was proposed for solving this problem.

5.2 Characteristics of the Fuzzy Controller

In this section we describe the main characteristics of the fuzzy controller for anesthesia control. We will define input and output variable of the fuzzy system. Also, the fuzzy rules of fuzzy controller previously designed will be described.

The fuzzy system is defined as follows:

1) Input variables: Blood pressure and Error
2) Output variable: Isoflurance concentration
3) Nine fuzzy if-then rules of the optimized system, which is the base for comparison
4) 12 fuzzy if-then rules of an initial system to begin the optimization cycle of the genetic algorithm.

The linguistic values used in the fuzzy rules are the following:

PB = Positive Big
PS = Positive Small
ZERO = zero
NB =Negative Big
NS = Negative Small

We show below a sample set of fuzzy rules that are used in the fuzzy inference system that is represented in the genetic algorithm for optimization.

if Blood pressure is NB and error is NB then conc_isoflurance is PS
if Blood pressures is PS then conc_isoflurance is NS
if Blood pressure is NB then conc_isoflurance is PB
if Blood pressure is PB then conc_isoflurance is NB
if Blood pressure is ZERO and error is ZERO then conc_isoflurance is ZERO
if Blood pressure is ZERO and error is PS then conc_isoflurance is NS
if Blood pressure is ZERO and error is NS then conc_isoflurance is PS
if error is NB then conc_isoflurance is PB
if error is PB then conc_isoflurance is NB
if error is PS then conc_isoflurance is NS
if Blood pressure is NS and error is ZERO then conc_isoflurance is NB
if Blood pressure is PS and error is ZERO then conc_isoflurance is PS.

5.3 Genetic Algorithm Specification

The general characteristics of the genetic algorithm that was used are the following:

NIND = 40; % Number of individuals in each subpopulation.
MAXGEN = 300; % Maximum number of generations allowed.

GGAP = .6; %"Generational gap", which is the percentage from the complete population of new individuals generated in each generation.

PRECI = 120; % Precision of binary representations.

SelCh = select('rws', Chrom, FitnV, GGAP); % Roulette wheel method for selecting the indivuals participating in the genetic operations.

SelCh = recombin('xovmp',SelCh,0.7); % Multi-point crossover as recombination method for the selected individuals.

ObjV = FuncionObjDifuso120_555(Chrom, sdifuso); Objective function is given by the error between the performance of the ideal control system given by the experts and the fuzzy control system given by the genetic algorithm.

5.4 Representation of the Chromosome

In Table 1 we show the chromosome representation, which has 120 binary positions. These positions are divided in two parts, the first one indicates the number of rules of the fuzzy inference system, and the second one is divided again into fuzzy rules to indicate which membership functions are active or inactive for the corresponding rule.

Table 1
Binary Chromosome Representation.

Bit assigned	Representation
1 a 12	Which rule is active or inactive
13 a 21	Membership functions active or inactive of rule 1
22 a 30	Membership functions active or inactive of rule 2
...	Membership functions active or inactive of rule...
112 a 120	Membership functions active or inactive of rule 12

6 Simulation Results

We describe in this section the simulation results that were achieved using the hierarchical genetic algorithm for the optimization of the fuzzy control system, for the case of anesthesia control. The genetic algorithm is able to evolve the topology of the fuzzy system for the particular application. We used 300 generations of 40 individuals each to achieve the minimum error. We show in Figure 7 the final results of the genetic algorithm, where the error has been minimized. This is the case in which only nine fuzzy rules are needed for the fuzzy controller. The value of the minimum error achieved with this particular fuzzy logic controller was of 0.0064064, which is considered a small number in this application.

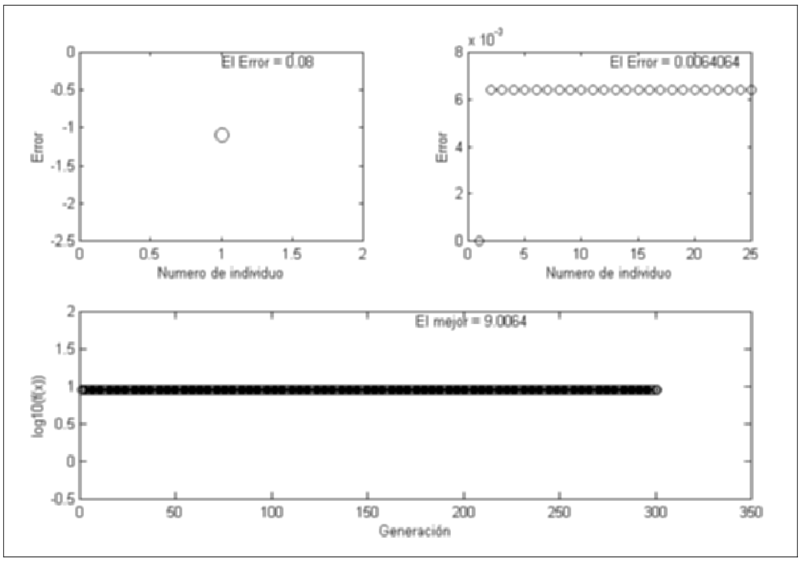


Figure 7 Plot of the error after 300 generations of the HGA.

In Figure 8 we show the simulation results of the fuzzy logic controller produced by the genetic algorithm after evolution. We used a sinusoidal input signal with unit amplitude and a frequency of 2 radians/second, with a transfer function of [1/(0.5s +1)]. In this figure we can appreciate the comparison of the outputs of both the ideal controller (1) and the fuzzy controller optimized by the genetic algorithm (2). From this figure it is clear that both controllers are very similar and as a consequence we can conclude that the genetic algorithm was able to optimize the performance of the fuzzy logic controller. We can also appreciate this fact more clearly in Figure 9, where we have amplified the simulation results from Figure 8 for a better view.

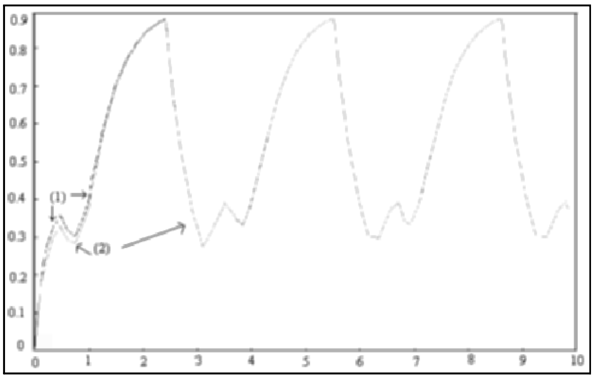


Figure 8 Comparison between outputs of the ideal controller (1) and the fuzzy controller produced with the HGA (2).

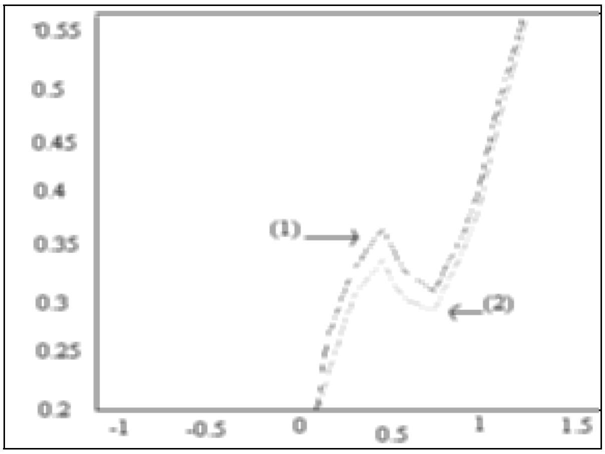


Figure 9 Zoom in of figure 8 to view in more detail the difference between the controllers.

Finally, we show in Figure 10 the block diagram of the implementation of both controllers in Simulink of MATLAB. With this implementation we are able to simulate both controllers and compare their performances.

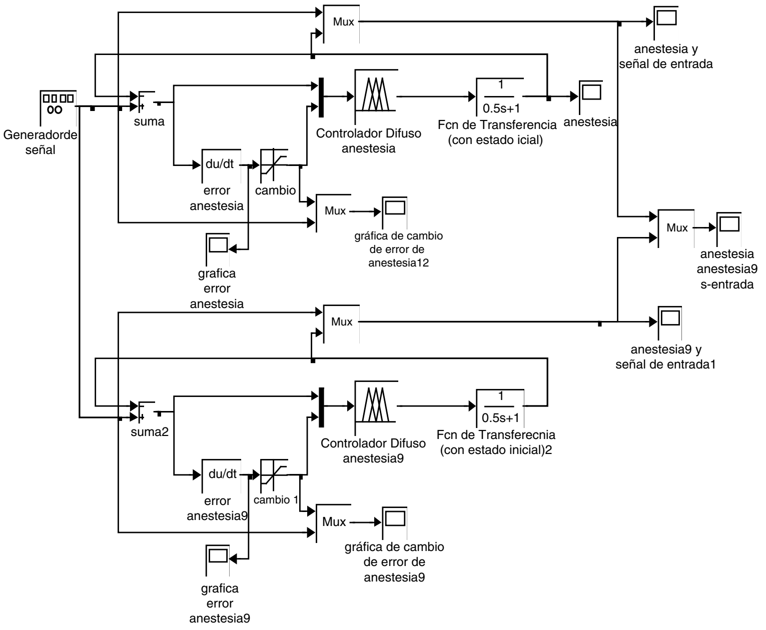


Figure 10 Implementation in Simulink of MATLAB of both controllers for comparison of their performance.

7 Conclusions

We consider in this paper the case of automatic anesthesia control in human patients for testing the optimized fuzzy controller. We did have, as a reference, the best fuzzy controller that was developed for the automatic anesthesia control (Karr and Gentry 1993, Lozano 2003), and we consider the optimization of this controller using the HGA approach. After applying the genetic algorithm the number of fuzzy rules was reduced from 12 to 9 with a similar performance of the fuzzy controller. Of course, the parameters of the membership functions were also tuned by the genetic algorithm. We did compare the simulation results of the optimized fuzzy controllers obtained with the HGA against the best fuzzy controller that was obtained previously with expert knowledge, and control is achieved in a similar fashion.

Acknowledgments

We would like to thank the Research Grant Committee of COSNET for the financial support given to this project (under grant 424.03-P). We would also like to thank CONACYT for the scholarships given to the students that work in this research project (Gabriel Huesca and Fevrier Valdez).

References

O. Castillo and P. Melin (2001), “Soft Computing for Control of Non-Linear Dynamical Systems”, Springer-Verlag, Heidelberg, Germany.

O. Castillo and P. Melin (2003), “Soft Computing and Fractal Theory for Intelligent Manufacturing”, Springer-Verlag, Heidelberg, Germany.

J. Holland, (1975), "Adaptation in natural and artificial systems" (University of Michigan Press).

A. Homaifar and E. McCormick (1995), “Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms”, *IEEE Trans. Fuzzy Systems*, vol. 3, pp. 129-139.

J.-S. R. Jang and C.-T. Sun (1995) “Neurofuzzy fuzzy modeling and control”, *Proc. IEEE*, vol. 83, pp. 378-406.

J.-S. R. Jang, C.-T. Sun, and E. Mizutani (1997), "Neuro-fuzzy and Soft Computing, A computational approach to learning and machine intelligence", , Prentice Hall, Upper Saddle River, NJ.

C.L. Karr and E.J. Gentry (1993), “Fuzzy control of pH using genetic algorithms”, *IEEE Trans. Fuzzy Systems*, vol. 1, pp. 46-53.

A. Lozano (2003), "Optimización de un Sistema de Control Difuso por medio de algoritmos genéticos jerarquicos", Thesis, Dept. of Computer Science, Tijuana Institute of Technology, Mexico.

K.F. Man, K.S. Tang, and S. Kwong (1999), "Genetic Algorithms: Concepts and Designs", Springer Verlag.

P. Melin and O. Castillo (2002), “Modelling, Simulation and Control of Non-Linear Dynamical Systems”, Taylor and Francis, London, Great Britain.

T.J. Procyk and E.M. Mamdani (1979), “A linguistic self-organizing process controller” *Automatica*, vol. 15, no. 1, pp 15-30.

K.-S. Tang, K.-F. Man, Z.-F. Liu and S. Kwong (1998), “Minimal fuzzy memberships and rules using hierarchical genetic algorithms”, *IEEE Trans. on Industrial Electronics*, vol. 45, no. 1.

J. Yen, and R. Langari (1999), "Fuzzy Logic: intelligence, control and information", Prentice Hall, Inc.

T. Yoshikawa, T. Furuhashi and Y. Uchikawa (1996), “Emergence of effective fuzzy rules for controlling mobile robots using NA coding method”, *Proc. ICEC’96*, Nagoya, Japan, pp. 581-586.

Part XVI

Recent Developments in Support Vector and Kernel Machines

Analyzing Magnification Factors and Principal Spread Directions in Manifold Learning

Junping Zhang[1], Li He[1,2], Zhi-Hua Zhou[3]

[1]Shanghai Key Laboratory of Intelligent Information Processing
Department of Computer Science and Engineering, Fudan University, Shanghai 200433, China
jpzhang@fudan.edu.cn
[2]Department of Mathematics, Fudan University, Shanghai 200433, China
demonstrate@163.com
[3]National Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China
zhouzh@nju.edu.cn

Summary. Great amount of data under varying intrinsic features is thought of as high dimensional nonlinear manifold in the observation space. How to analyze the mapping relationship between the high dimensional manifold and the corresponding intrinsic low dimensional one quantitatively is important to machine learning and cognitive science. In this paper, we propose SVD (singular value decomposition) based magnification factors and spread direction for quantitative analyzing the relationship. The result of conducting experiments on several databases show the advantages of this proposed SVD-based approach in manifold learning.

Key words: SVD-based Magnification Factors, SVD-based Principal Spread Directions, Manifold Learning, Locally Linear Embedding Algorithm

1 Introduction

The subspace learning based approach (Turn and Pentland 1991, Nayar et al. 1995) has achieved significant success in pattern recognition and computer vision. While current algorithms deal with linear subspaces such as using principal component analysis (PCA)(Turn and Pentland 1991), images of an object under varying illumination, pose, and non-rigid deformation are empirically thought to constitute highly nonlinear manifold in the observation space, as generated by intrinsic variables. However, these linear methods are bound to ignore subtleties of manifolds such as concavities and protrusions, which is bottleneck to achieve more accurate recognition. This problem should be solved before we can reach high performance recognition system.

It is shown empirically that image manifolds may have large curvature (Hau-Minn et al. 1998, Seung and Daniel 2000). We therefore assume that an effective, high performance object recognition system should be based on the manifolds of the objects. Meanwhile, the quantitative analysis of mapping relationship from intrinsic low-dimensional manifold to corresponding high-dimensional manifold and manifold-based matching are the keys to the success.

During these years progresses have been achieved in modelling nonlinear subspaces or manifolds. Rich literature exists on manifold learning. On the basis of different representations of manifold learning, this can be roughly classified into four major classes: projection methods, generative methods, embedding methods, and mutual information methods.

1. The first is to find principal surfaces passing through the middle of data, such as the principal curves (Hastie and Stuetzle 1988, Kégl et al. 2000). Though geometrically intuitive, this one has difficulty in generalizing the global variable–arc-length parameter– into higher dimensional surface.
2. The second adopts generative topology models (Bishop et al. 1998, Chang and Ghosh 2001, Smola et al. 1999), and hypothesizes that observed data is generated from the evenly spaced low-dimensional latent nodes. And then the mapping relationship between the observation space and the latent space can be modeled. Resulting from the inherent insufficiency of the adopted EM (Expectation-Maximization) algorithms, nevertheless, the generative models fall into local minimum easily and have slow convergence rates.
3. The third is generally divided into global and local embedding algorithms. ISOMAP (Tenenbaum et al. 2000), as a global algorithm, presumes that isometric properties should be preserved in both the observation space and the intrinsic embedding space in the affine sense. And extension to conformal mappings is also discussed in (Silva and Tenenbaum 2002). On the other hand, Locally Linear Embedding (LLE) (Roweis and Lawrance 2000) and Laplacian Eigenamp (Belkin and Niyogi 2003) focus on the preservation of local neighbor structure.
4. In the fourth category, it is assumed that the mutual information is a measurement of the differences of probability distribution between the observed space and the embedded space, as in stochastic nearest neighborhood (henceforth SNE) (Hinton and Roweis 2003) and manifold charting (Brand and MERL 2003).

While there are many impressive results on the qualitative representation (such as discovering expression and pose of face images by user) of the intrinsic features of a manifold, there are fewer reports on the quantitative analysis of manifold learning. Therefore, we present SVD-based approach for analyzing the mapping relationship from intrinsic low-dimensional manifold to high-dimensional manifold. Several principal parameters such as magnification factors and principal spread direction are studied. Magnification factor is the ratio of local area in the low-dimensional space and the corresponding area in the high-dimensional space. Principal spread directions reflect the major variation axis of local area from the low-dimensional space to high-dimensional space. Investigation on these mapping relationships are of considerable interest in both neuro-biological and data analysis contexts (Bishop et al. 1997).

The paper is organized as follows: After given survey on manifold learning in the first section, one of manifold learning algorithms (Locally linear embedding algorithm) is briefly introduced in section 2. In section 3, SVD-based magnification factors and principal spread directions is presented. In the penultimate section encouraging result from the experiments in several database show its advantage and the final section we discuss some potential problems and conclude the paper.

2 Dimensionality Reduction

To establish the mapping relationship between the observed data and the corresponding low-dimensional data, many manifold learning algorithms can be used. In this paper, we focus our study on the locally linear embedding (LLE) algorithm (Roweis and Lawrance 2000) to obtain the corresponding low-dimensional data Y ($Y \subset \mathbb{R}^d$) of the training set X ($X \subset \mathbb{R}^N, N \gg d$). And then the dataset (X, Y) is used for analyzing the mapping relationship.

The main principle of LLE algorithm is to preserve local order relation of data in both the embedding space and the intrinsic space. Each sample in the observation space is a linearly weighted average of its neighbors. The basic LLE algorithm can be described as follows:

Step 1: Define

$$\psi(W) = \parallel x_i - \sum_{j=1}^{K} W_{ij} x_{ij} \parallel^2 \tag{1}$$

Where samples x_{ij} are the neighbors of x_i. Considering the constraint term $\sum_{j=1} W_{ij} = 1$, and if x_i and x_{ij} are not in the same neighbor, $W_{ij} = 0$. Therefore, the sum of neighbor samples in Formula (1) can be computed from 1 to K. And the weighted matrix W is calculated according to the classical least square approach.

Step 2: Define

$$\varphi(Y) = \parallel y_i - \sum_{j=1}^{K} W^*_{ij} y_{ij} \parallel^2 \tag{2}$$

where $W^* = \arg\min_W \psi(W)$. Step 2 of the algorithm is equivalent to approximate the nonlinear manifold around point y_i by the linear hyperplane that passes through its neighbors $\{y_{i1}, \ldots, y_{ik}\}$. Considering that the objective $\varphi(Y)$ is invariant to translation in Y, constraint term $\sum_i y_i = 0$ is added in the step 2. Moreover, the other term $\sum_i y_i y_i^T / n = I$ is to avoid the degenerate solution of $Y = 0$. Hence, step 2 reduces to an eigenvector decomposition problem as follows:

$$\begin{aligned}
Y^* &= \arg\min_Y \varphi(Y) \\
&= \| \; y_i - \sum_{j=1}^{K} W^*_{ij} y_{ij} \; \|^2 \\
&= \arg\min_Y \| \; (I - W)Y \; \|^2 \\
&= \arg\min_Y Y^T (I - W)^T (I - W)Y
\end{aligned} \tag{3}$$

The optimal solution of Y^* in Formula (3) is the smallest eigenvectors of matrix $(I - W)^T(I - W)$. Obviously, those eigenvalues which are zero need be removed. So we need to compute the bottom $(d + 1)$ eigenvectors of the matrix and discard the smallest eigenvector considering constraint term.

Thus, we obtain the corresponding low-dimensional data set Y in embedding space. And the completed set (X, Y) is used for the subsequent quantitative analysis of intrinsic features.

3 SVD-based Magnification Factors and Principal Spread Directions

Manifold learning algorithms make impressive results on the discovery of the principal intrinsic features of high-dimensional data (for example, pose and expression of face images), the evaluation of the final results is nevertheless qualitative and the effectiveness depends on the experiences of researchers themselves. In this section, the principal objective is to analyze the mapping relationship objectively, such as the magnification factors and principal spread directions. The concept of magnification factors is relate to the topographic maps in the brain, in which neuro-psychologists found that those in the visual and somatosensory areas of the cortex relate the two-dimensional spatial density of sensors to the two-dimensional spatial density of the corresponding cortical cells (Bishop et al. 1997). The research on how high-dimensional nonlinear manifold is generated from intrinsic independent features has thus potential value for machine learning and cognitive sciences. While Bishop had explored these properties of generative topographic mapping ever, how to quantitatively analyze the effectiveness of manifold learning algorithms has not been studied (Bishop et al. 1998).

In this paper, we hence propose a new SVD-based approach to overcome the disadvantage of present manifold learning algorithms to quantitatively analyze of the relationship between intrinsic features and high-dimensional data. It is worth noting that $(x_i^1, \cdots, x_i^n)$ represents the ith sample and the superscript means the dimension or attribute of sample in the section.

To realize quantitative analysis, the main difficulty is that the mapping relationship obtained by the K-nearest neighbors (basic parameter in the manifold learning algorithms) is undifferentiate. Therefore, an alternative RBF model which has the ability of universal approximation is introduced as follows:

$$x_i = \sum_{j=1}^{n} w_i k(y_i, y_j) \tag{4}$$

where y_i is the corresponding low-dimensional point of sample x_i. Without loss of generality, gaussian kernel

$$k(y_i, y_j) = k(y_j, y_i) = \exp^{-\frac{\|y_i - y_j\|^2}{2\sigma^2}} \tag{5}$$

is used. For simplicity, the corresponding matrix form of Eq. (4) is:

$$X = WK \tag{6}$$

where $W \in \mathbf{M}_{N,n}, K \in \mathbf{M}_{n,n}$. $\mathbf{M}_{s,t}$ means that the matrix has s rows and t columns, and W and K is formulated as follows:

$$W = (w_1, \cdots, w_n) \quad K = \begin{pmatrix} k(y_1, y_1) & \cdots & k(y_1, y_n) \\ \cdots & \cdots & \cdots \\ k(y_n, y_1) & \cdots & k(y_n, y_n) \end{pmatrix} \tag{7}$$

Because the completed data (X, Y) has been obtained by mentioned LLE algorithm, the matrix W is calculated as follows:

$$W = XK^{-1} \tag{8}$$

where K^{-1} means Moore-Penrose inverse matrix of K.

Obviously, parameter σ^2 have a critical influence on the final experimental result. In this paper, parameter σ^2 is heuristical selected based on the number of samples and K-nearest neighbors LLE used. Considering geometric properties, we first compute the range from the maximum norm to minimum norm of samples Y under the low-dimensional space. Then let the range be evenly partitioned and the number of partitions is that the number of samples divides neighbor factor K. Finally, σ^2 is equal to the ratio of the range to the number of partitions.

3.1 Magnification Factors

To quantitatively analyze the effectiveness of manifold learning algorithms, we first focus our research on the magnification factors from each low-dimensional sample to corresponding high-dimensional sample.

Let $x_i \in \mathbb{R}^N$, the partial derivation $\Delta^j x \in \mathbb{R}^N$ with respect to the jth dimension is formulated as follows:

$$\Delta^j(x) = \frac{\partial x}{\partial y^j}\Delta y^j \in \mathbf{M}_{N,1} \tag{9}$$

where

$$\frac{\partial x}{\partial y^j} = \psi_j W^T \tag{10}$$

and $\psi_j \in \mathbf{M}_{n,1}$. Because $\psi_j = (\psi_{1,j}, \cdots, \psi_{n,j})^T$, $\psi_{l,j}$ is therefore calculated as follows:

$$\psi_{l,j} = -\frac{k(y, y_l)(y^j - y_l^j)}{\sigma^2} \quad l = 1, \cdots, n \tag{11}$$

Consider Eq. (11), the Jacobi matrix is represented as follows:

$$J = (\frac{\partial x}{\partial y^1}, \cdots, \frac{\partial x}{\partial y^d}) = -\frac{1}{\sigma^2} W \begin{pmatrix} k(y, y_1)(y - y_1)^T \\ \vdots \\ k(y, y_n)(y - y_n)^T \end{pmatrix} \in \mathbf{M}_{N,d} \tag{12}$$

To visualize the high-dimensional manifold, we always reduce the dimensions of the manifold into 2 or 3. Assuming the intrinsic dimension d is equal to 2, the magnification factors are thus approximated by the ratio of the area A_x of high-dimensional space and the area A_y in the infinitesimal and local sense. That is to say, the magnification factors can be calculated on the basis of Eq. (13) through Eq. (15):

$$\begin{aligned} A_x^2 &= \|\Delta^1 x\|^2 \|\Delta^2 x\|^2 - ((\Delta^1 x)^T (\Delta^2 x))^2 \\ &= \|\Delta y^1 \frac{\partial x}{\partial y^1}\|^2 \|\Delta y^2 \frac{\partial x}{\partial y^2}\|^2 - ((\Delta y^1 \frac{\partial x}{\partial y^1})^T (\Delta y^2 \frac{\partial x}{\partial y^2}))^2 \end{aligned} \tag{13}$$

$$A_y^2 = (\Delta y^1 \Delta y^2)^2 \tag{14}$$

$$\frac{A_x}{A_y} = \sqrt{\|\frac{\partial x}{\partial y^1}\|^2 \|\frac{\partial x}{\partial y^2}\|^2 - ((\frac{\partial x}{\partial y^1})^T (\frac{\partial x}{\partial y^2}))^2} \tag{15}$$

With respect to the complete data (X, Y), the magnification factor of sample $y_l, l = 1, \cdots, n$ can be calculated as follows:

$$\frac{A_x}{A_y}|_{y=y_l} = \sqrt{\|\frac{\partial x}{\partial y^1}|_{y=y_l}\|^2 \|\frac{\partial x}{\partial y^2}|_{y=y_l}\|^2 - ((\frac{\partial x}{\partial y^1}|_{y=y_l})^T (\frac{\partial x}{\partial y^2}|_{y=y_l}))^2} \tag{16}$$

And the corresponding Jacobi matrix of Eq. (12) is re-written as follows:

$$J = (\frac{\partial x}{\partial y^1}, \frac{\partial x}{\partial y^2}) = -\frac{1}{\sigma^2} W \begin{pmatrix} k(y, y_1)(y^1 - y_1^1) & k(y, y_1)(y^2 - y_1^2) \\ \vdots & \vdots \\ k(y, y_n)(y^1 - y_n^1) & k(y, y_n)(y^2 - y_n^2) \end{pmatrix} \in \mathbf{M}_{N,2} \tag{17}$$

Thus the magnification factors in the 2-dim intrinsic space are computed as follows:

$$\frac{A_x}{A_y} = \sqrt{\det(J^T J)} \tag{18}$$

With respect to Eq. (18), we can perform quantitative analysis of magnification factors of all sample.

3.2 Principal Spread Directions

Another objective of the paper is to compute principal spread directions from data in the low-dimensional manifold to that of high-dimensional nonlinear manifold in the local sense. From mathematical view of point, finding the principal spread directions is to compute the eigenvalues and eigenvectors based on outer-matrix of data (Bishop et al. 1998). The outer-matrix is computed by the following formula:

$$\sum_{i=1}^{d} \frac{\partial x}{\partial y^l}|_{P_x}^T \frac{\partial x}{\partial y^l}|_{P_x} \tag{19}$$

It is not difficult to show that Eq. (19) is equivalent to $J^T J$. Thus the eigenvectors $\hat{x}^l$ and the corresponding eigenvalues λ_l can be calculated:

$$\hat{x}^l J^T J = \lambda_l \hat{x}^l \quad l = 1, 2, \cdots, d \tag{20}$$

In the differential and local sense, we can assume that high-dimensional manifold is linearly generated from low-dimensional manifold. Namely, $x = yJ$. Therefore substitute the equation into Eq. (20), we have:

$$\hat{y} J J^T J = \lambda_l \hat{y}^l J \tag{21}$$

Since J is an $N \times d$ matrix, and rank is d and positive inverse, therefore Eq. (21) has refined representation:

$$\hat{y} J J^T = \lambda_l \hat{y}^l \tag{22}$$

It is obvious that with the computation of spectral properties of Eq. (22), the principal spread direction based on the eigenvalues and eigenvectors can be effective obtained for the quantitative analysis of manifold learning algorithms.

3.3 The Proposed Approach

Although the above-mentioned eigenvalues and eigenvectors help us to analyze the relationship between low-dimensional manifold and high-dimensional manifold, the computation of eigenvalue and eigenvector of JJ^T is ineffective. As we know, $J \in \mathbf{M}_{N,d}$, then $JJ^T \in \mathbf{M}_{N,N}$. If let N=4000 and each attribute be represented with 8-bits bytes, for example, then the storing cost of JJ^T is:

$$4000 \times 4000 \times 8\text{bytes} \times \frac{1\text{Mb}}{1024 \times 1024\text{bytes}} \approx 120\text{Mb} \tag{23}$$

Another disadvantage is that the number of eigenvalues computed with JJ^T is generally larger than that might have been anticipated.

We thus propose to use a SVD (Singular Value Decomposition)-based approach to overcome the disadvantages of computing outer-matrix. Let

$$J = P_1 \Sigma P_2 \tag{24}$$

where $P_1 \in \mathbf{M}_{N,N}, \Sigma \in \mathbf{M}_{N,d}, P_2 \in \mathbf{M}_{d,d}$, and meanwhile P_1, P_2 are orthogonal matrices, $\Sigma = \mathrm{diag}\{\sigma_1, \sigma_2\}$, therefore:

$$JJ^T = P_1 \Sigma P_2 (P_1 \Sigma P_2)^T = P_1 \Sigma P_2 P_2^{-1} \Sigma^T P_1^{-1} = P_1 \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix} P_1^{-1} \tag{25}$$

It is not difficult to show that when $d = 2$, the length r_i of principal spread directions is equal to σ_i, and the principal spread directions are P_1. Actually, the magnification factors can be calculated based on SVD approach as follows:

$$\frac{A_x}{A_y} = \sqrt{\det(J^T J)} = \sqrt{\det(P_2^{-1} \begin{pmatrix} \sigma_1^2 & \\ & \sigma_2^2 \end{pmatrix} P_2)} = \sigma_1 \sigma_2 \tag{26}$$

With the proposed SVD-based approach, the computation of the magnification factors and principal spread directions is realized with one step only. Meanwhile, we avoid the disadvantage of storing large-scale matrix JJ^T.

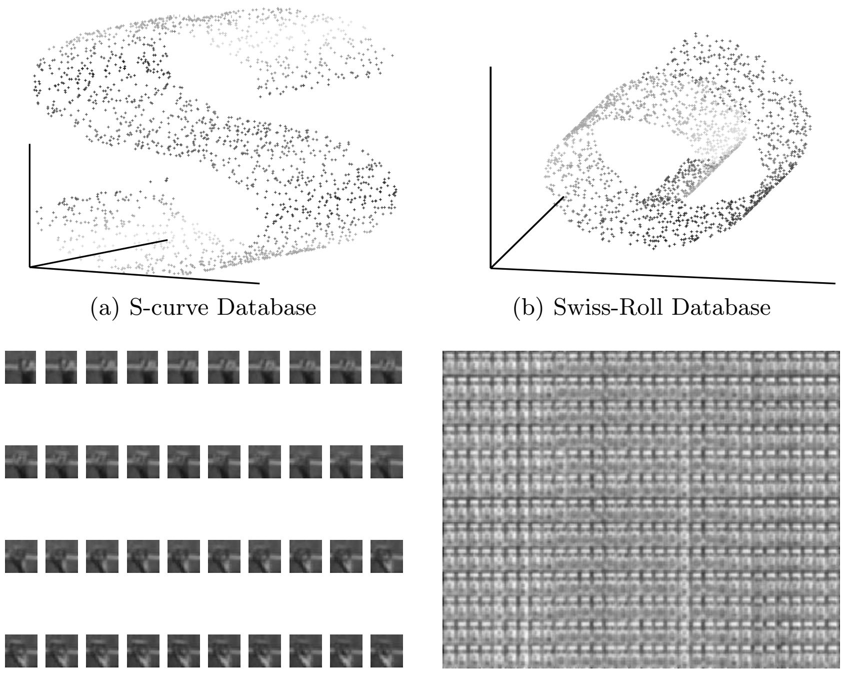
4 Experiments

Experiments both in artificial databases (Scurve and Swissroll manifolds) and practical databases (hand-turning-cup and Frey face databases) are used to verify the feasibility of the proposed SVD-based approach. The examples of these databases can be illustrated in Figure 1.

Two artificial databases in the 3-dimensional space are generated by two-dimensional plane. As for 'hand rolling cup" database[1], the original pixel-based dimensions of the observation space are 3840 dimensions (64*60 pixels), whereas the known intrinsic degree of freedom is of only one-dimension, namely, horizontal rotation. And Frey face database are 560 dimensions (20*28 pixels, intensity 256, 1956 samples). These images are regarded as points in the high-dimensional vector space, with each input dimension corresponding to the brightness of one pixel.

Considering the demand of visualization, the sizes of samples in four databases have been reduced through vector quantization algorithm. For example, the number of samples in Frey database has been reduced 218 and the number of samples in Hand databases is 96. Meanwhile, the scopes of magnification factors of samples are normalized to gray levels [0, 255] so that

[1] http://vasc.ri.cmu.edu/idb/html/motion/hand/index.html



(a) S-curve Database

(b) Swiss-Roll Database

(c) Hand-turning-Cup Database

(d) Frey Face Database

Fig. 1. Examples of Several Databases

the 255th gray level corresponds to the largest magnification factors, and vice versa. Also we visualize the magnification factors with the size of the radius of each sample. The bigger the magnification factor is, the brighter each sample is, and thus the larger the radius is. Moreover, we assume that the intrinsic dimensions of these databases are 2 and dimensionality reduction is realized with LLE algorithm. Based on the idea of local linearity, neighbor factor $K = 8$ except for the hand-turning-cup database (where K=4). The quantitative analysis is thus performed in the 2-dim spaces. The corresponding results based on the proposed SVD-based approach are illustrated in Figure 2 through Figure 7. And sample in both Figure 5 and Figure 7 is the corresponding high-dimensional image of each low-dimensional sample in hand-turning-cup database and Frey face database, respectively. (Note: These results can be seen clearer in the PDF files)

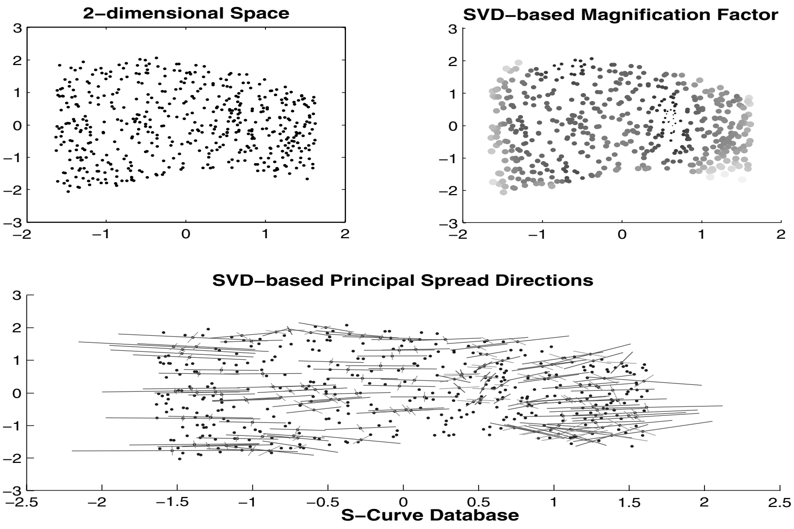


Fig. 2. S-curve Database

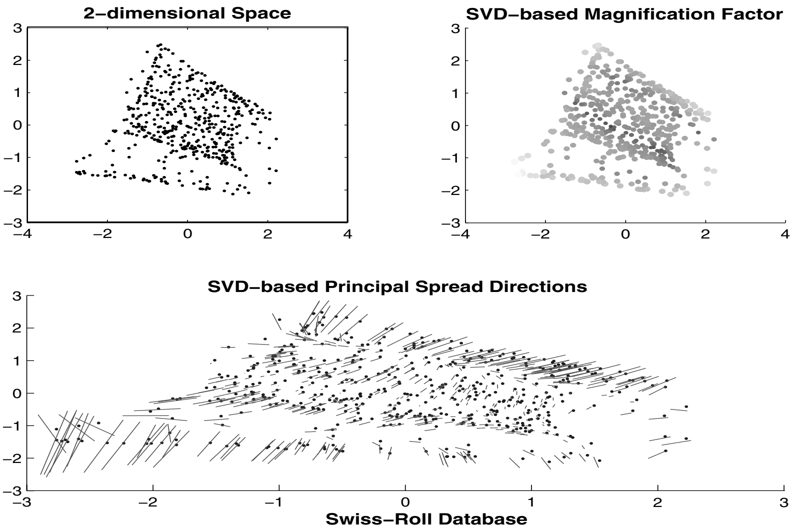


Fig. 3. Swiss-roll Database

In all the four Figures on principal spread directions, the longer line of each sample represents the first spread direction, and the second line is orthogonal to the first spread direction, respectively. From the Figures we can find several major properties:

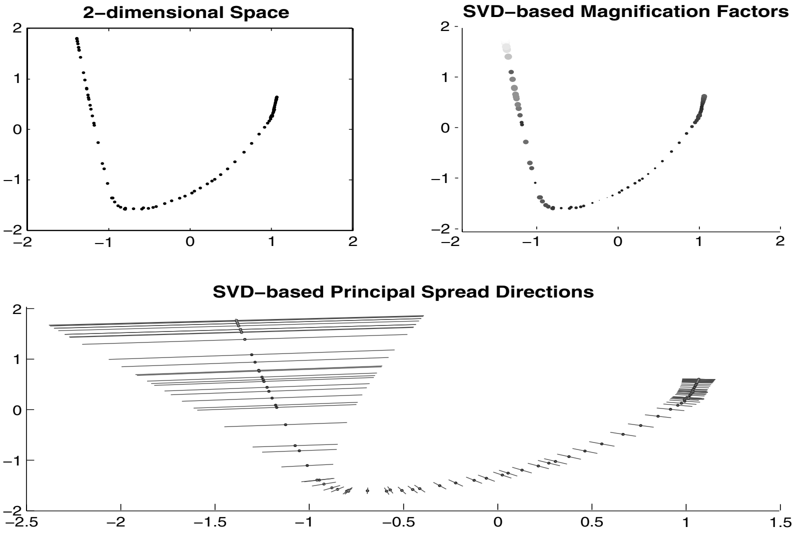


Fig. 4. Hand-Turning-Cup Database

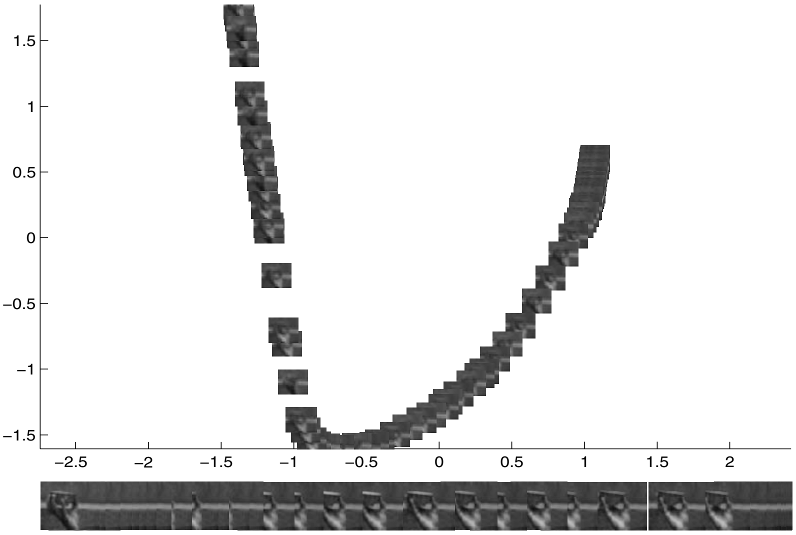


Fig. 5. Hand-Turning-Cup Database

- Both magnification factors and principal spread directions of samples in the boundary than that of samples in the centers have stronger and larger variation trend, such as Figure 2 and Figure 3.
- Each sample and its neighbors have similar magnification factors and principal spread directions.

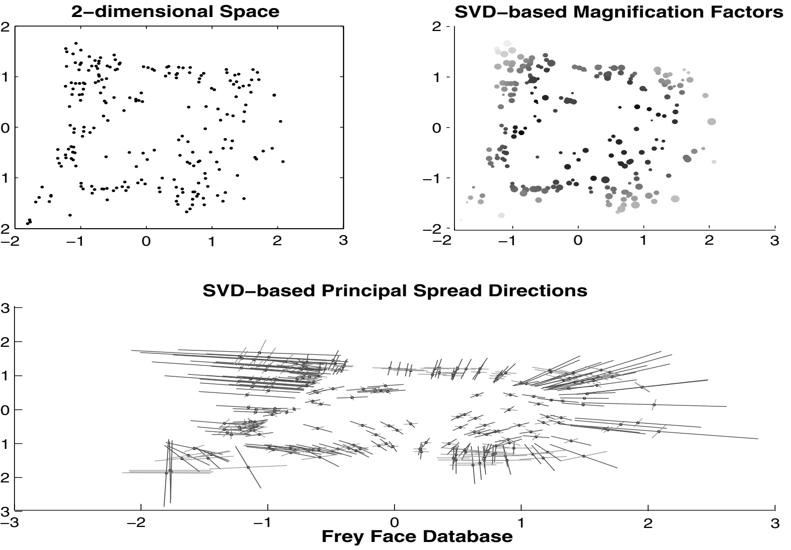


Fig. 6. Frey Face Database

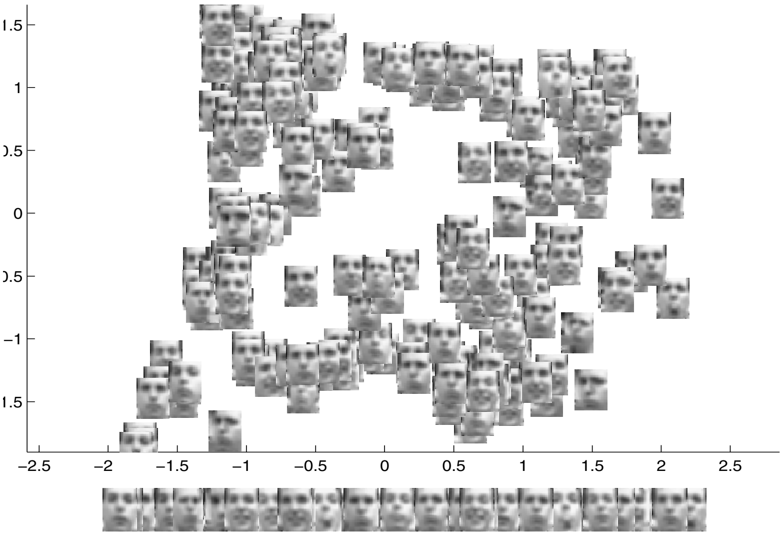


Fig. 7. Frey Face Database

- Comparing these four Figures with the original Figures, it is easy to see that the higher the curvature of high-dimensional manifold has, the more dramatic the principal spread directions change.
- Furthermore, if the original dimensions are 3, for example, S-curve and Swiss-roll database, the principal spread directions has clear regularity which can reflect the mapping relationship from the 2-dim space to 3-dim

space. If the original dimensions are higher, the intrinsic regularities to be extracted sound more complexity. We assume that these databases may have other unknown intrinsic variables which lead to the generation of high-dimensional nonlinear manifold.

As a result, we quantitatively analyze the mapping relationship between high-dimensional manifold and corresponding low-dimensional manifold with the proposed SVD-based approach. Also the figures provide more intrinsic information of data comparing with LLE algorithm shows. Therefore, the proposed approach may aim us to analyze high-dimensional data more effectively.

5 Conclusions

In this paper, we propose SVD based approach for quantitatively analyzing magnification factors and principal spread directions of high dimensional data. The proposed SVD based approach can be a complementary tool to aid researcher to evaluate the feasible of manifold learning algorithms effectively and has potential applications in cognitive sciences and other scientific fields. In fact, we also investigate the classification ability of manifold learning. Due to the limitation of size, we have to drop the section. The details on how to achieve recognition task based on manifold learning approach can be seen in (Junping et al. 2004a, Junping et al. 2004b).

At the same time, several problems remain to be solved in future works. In the paper, the selection of parameter σ^2 in the proposed SVD based approach depends on heuristical approach. So further study could be in how to choose the parameter automatically. While we focus our research on Locally Linear Embedding algorithm, in addition, it is easy to generalize the proposed approach to other manifold learning algorithms such as ISOMAP (Tenenbaum et al. 2000), Laplacian Eigenmap (Belkin and Niyogi 2003). The corresponding experimental results will be presented in a longer version of this paper.

Acknowledgements

The authors would like to thank the anonymous reviewers for their detailed comments and questions which improved the quality of the presentation of this paper. This work was partially supported by 2003-HT-FD05 and the National Outstanding Youth Foundation of China under the Grant No. 60325207.

References

Mikhail Belkin and Partha Niyogi, (2003), “Laplacian Eigenmaps for Dimensionality Reduction and Data Representation,” *Neural Computation*, vol. 15, Issue 6, pp. 1373-1396.

C. M. Bishop, M. Svensén, and C. K. I. Williams. (1997), "Magnification Factors for the GTM Algorithm," In *Proceedings of the IEE Fifth International Conference on Artificial Neural Networks*, pages 64-69, London, 1997c, IEE.

C. M. Bishop, M. Sevensén, and C. K. I. Williams, (1998), "GTM:The generative topographic mapping," *Neural Computation*, 10, pp. 215-234.

M. Brand, MERL, (2003), "Charting a manifold," *Neural Information Processing Systems 15 (NIPS'02).*

K. Chang and J. Ghosh, "A unified model for probabilistic principal surfaces," (2001), *IEEE transactions on Pattern Analysis and Machine Intelligence*, 23(1), pp. 22-41.

T. Hastie and W. Stuetzle, (1988), "Principla Curves,"*Journal of the American Statistical Association*, 84(406), pp. 502-516.

Haw-Minn Lu, Yeshaiahu Fainman, and Robert Hecht-Nieslen, (1998), "Image Manifolds," *in Proc. SPIE*, vol. 3307, pp.52-63.

G. Hinton and S. Roweis, (2003), " Stochastic Neighbor Embedding," *Neural Information Processing Systems 15 (NIPS'02)*, pp. 833-840.

B. Kégl, A. Krzyzak, T. Linder, and K. Zeger, (2000), "Learning and design of principal curves,"*IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 3, pp. 281-297.

Shree K. Nayar, Sameer A. Nene, Hiroshi Murase, (1995), "Subspace Methods for Robot Vision," Technical Report CUCS-06-95, Columbia University, New York.

S. T. Roweis, and K. S. Lawrance, (2000), "Nonlinear Dimensionality reduction by locally linear embedding," *Science*, 290, pp. 2323-2326.

H. S. Seung, and D. L. Daniel, (2000), "The Manifold Ways of Perception," *Science*, 12, pp.2268-2269.

V. S. Silva, J. B. Tenenbaum, (2002), "Unsupervised Learning of curved manifolds," *Nonlinear Estimation and Classification*, Springer-Verlag, New York.

A. J. Smola, S. Mika, et al., (1999), "Regularized Principal Manifolds,"*In Computational Learning Theory: 4th European Conference*, Vol 1572 of Lecture Notes in Artificial Intelligence, New York: Springer, pp. 251-256.

Joshua. B. Tenenbaum, Vin de Silva, and John C. Langford, (2000), "A global geometric framework for nonlinear dimensionality reduction,"*Science*, 290, pp. 2319-2323.

M. Turk and A. Pentland, (1991), "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience*, vol 3, no.1, pp.71-86.

Junping Zhang, Stan Z. Li, and Jue Wang, (2004a), "Nearest Manifold Approach for Face Recognition," *The 6th international Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, May.

Junping Zhang, Stan Z. Li, and Jue Wang. (2004b), "Manifold Learning and Applications in Recognition," in *Intelligent Multimedia Processing with Soft Computing.* Yap Peng Tan, Kim Hui Yap, Lipo Wang (Ed.), Springer-Verlag, Heidelberg.

Bag Classification Using Support Vector Machines

Uri Kartoun, Helman Stern, Yael Edan

{kartoun|helman|yael}@bgu.ac.il
Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Be'er-Sheva, Israel

Abstract: This paper describes the design of multi-category support vector machines (SVMs) for classification of bags. To train and test the SVMs a collection of 120 images of different types of bags were used (backpacks, small shoulder bags, plastic flexible bags, and small briefcases). Tests were conducted to establish the best polynomial and Gaussian RBF (radial basis function) kernels. As it is well known that SVMs are sensitive to the number of features in pattern classification applications, the performance of the SVMs as a function of the number and type of features was also studied. Our goal here, in feature selection is to obtain a smaller set of features that accurately represent the original set. A K-fold cross validation procedure with three subsets was applied to assure reliability. In a kernel optimization experiment using nine popular shape features (area, bounding box ratio, major axis length, minor axis length, eccentricity, equivalent diameter, extent, roundness and convex perimeter), a classification rate of 95% was achieved using a polynomial kernel with degree six, and a classification rate of 90% was achieved using a RBF kernel with 27 sigma. To improve these results a feature selection procedure was performed. Using the optimal feature set, comprised of bounding box ratio, major axis length, extent and roundness, resulted in a classification rate of 96.25% using a polynomial kernel with degree of nine. The collinearity between the features was confirmed using principle component analysis, where a reduction to four components accounted for 99.3% of the variation for each of the bag types.

Keywords: Support vector machines, multi-category classification, feature selection

1 Introduction

SVMs have emerged as very successful pattern recognition methods in recent years [16]. SVMs have yielded superior performance in various applications such as; text categorization [15], and face detection [12], content-based image retrieval [6], and learning image similarity [4].

The motivation here is detection of suspicious bags in a security situation. The usual method for bomb personnel is to blow up a suspicious bag, and any explosives contained therein. However, if the bag contains chemical, biological or radiological canisters, this may lead to disastrous results. Furthermore, the "blow-up" method also destroys important clues such as fingerprints, type of explosive, detonators and other signatures of importance for forensic analysis. Extraction of the bag contents using telerobotics avoids these problems [8]. In a telerobotic system, it is advantageous to automate bag classification which is coupled to robotic tactics such as shaking out of the bags contents. For cooperative human-robot interaction, in situations when autonomous capabilities fail, a human may be called in to distinguish the bag type. Here we focus on the autonomous operation of such a telerobotic system. In the autonomous mode, bags are classified by type using SVMs for the purpose of identifying initial manipulator grasp points. One side of the gripper must slide under the bag, and because of slippage, the grasp may fail. Thus, it is of importance to realize the type of bag, and the location of the opening before the grasp point assessment is made. Moreover, because we are dealing with a soft object here, with an unknown skin texture and unknown objects inside of the bag grasp may fail. Also, for each bag type a different rule set is evoked to determine the robot arm shake trajectories to discharge the contents of the bag for subsequent inspection.

In this paper we report on multi-category support vector classification of bags. The classification approach is presented in Section 2. Image processing operations are described in Section 3. Kernel optimization and optimal feature selection experimental results are described in Section 4. In section 5 an analysis of the results is given. The paper ends with conclusions and some directions for future work.

2 Classification Approach

SVMs belong to the class of maximum margin classifiers [5]. The goal of maximum margin classification in binary SVM classification [16] is to separate the two classes by a hyperplane such that the distance to the support vectors is maximized. SVMs perform pattern recognition between two classes by finding a decision surface that has maximum distance to the closest points in the training set which are termed support vectors. The procedure starts with a training set of points $x_i \in \Re n$, $i = 1, 2, \ldots, N$ where each point x_i belongs to one of two classes identified by the label $y_i \in \{-1,1\}$. This hyperplane is called the optimal separating hyperplane (OSH). The OSH has the form:

$$f(x) = \sum_{i=1}^{N} \alpha_i y_i x_i \cdot x + b. \tag{2.1}$$

The coefficients α_i and b in (2.1) are the solutions of a quadratic programming problem [16]. A data point x is classified by the sign of the right side of (2.1). For multi-category classification, (2.2) is used.

$$d(x) = \frac{\sum_{i=1}^{N} \alpha_i y_i x_i \cdot x + b}{\| \sum_{i=1}^{N} \alpha_i y_i x_i \|}. \tag{2.2}$$

The sign of d is the classification result for x, and $| d |$ is the distance from x to the hyperplane. Intuitively, the farther away a point is from the decision surface, i.e., the larger $| d |$, the more reliable the classification result. The entire construction can be extended to the case of nonlinear separating surfaces. Each point x in the input space is mapped to a point $z = \Phi(x)$ of a higher dimensional space, called the feature space, where the data are separated by a hyperplane. The key property in this construction is that the mapping $\Phi(\cdot)$ is subject to the condition that the dot product of two points in the feature space $\Phi(x) \cdot \Phi(y)$ can be rewritten as a kernel function $\mathrm{K}(x, y)$. The decision surface has the equation:

$$f(x) = \sum_{i=1}^{N} \alpha_i y_i K(x,x_i) + b, \tag{2.3}$$

where, the coefficients $_i$ and b are solutions of a quadratic programming problem. Note, that $f(x)$ does not depend on the dimensionality of the feature space. Kernel functions commonly used for pattern recognition problems are polynomial of degree d (2.4) and gaussian radial basis (2.5).

$$K(x, y) = (1 + x \cdot y)^d, \tag{2.4}$$

$$K(x, y) = e^{-\frac{\|x-y\|^2}{2\sigma^2}}. \tag{2.5}$$

The dual category SVM classification was extended to multi-category classification by [1]. There are two basic strategies for solving q-class problems with SVMs. In the one-vs-all approach, q SVMs are trained. Each of the SVMs separates a single class from all remaining classes [14, 2]. In the pairwise approach (used in this work), $q(q\text{-}1)/2$ machines are trained. Each SVM separates a pair of classes. The pairwise classifiers are arranged in trees, where each tree node represents a SVM. Regarding the training effort, the one-vs-all approach is preferable since only q SVMs have to be trained compared to $q(q\text{-}1)/2$ SVMs in the pairwise approach. The run-time complexity of the two strategies is similar: the one-vs-all and the pairwise approaches require the evaluation of q and q-1

SVMs, respectively. Results on person recognition indicate similar classification performance for the two strategies [11]. The input to the SVMs is a set of features obtained from the bag image. The image processing feature extraction methods are described in the next section.

3 Image Processing

Image processing starts with a 24-bit color image of the robotic scene that contains a bag located on a platform. Four different bag types are considered: backpacks, small shoulder bags, plastic flexible bags, and small briefcases (Fig. 3.1). Image processing operations used are [7]: conversion to gray-scale, thresholding using the Otsu method [13], removal of noise from the image due to optical lens distortion, adaptation to ambient and external lighting conditions, and segmentation of the bag from the background. Using the MATLAB's Image Processing Toolbox [10], nine popular shape features were extracted from each segmented bag image: Area, Bounding Box Ratio, Major Axis Length, Minor Axis Length, Eccentricity, Equivalent Diameter, Extent, Roundness, Convex Perimeter.

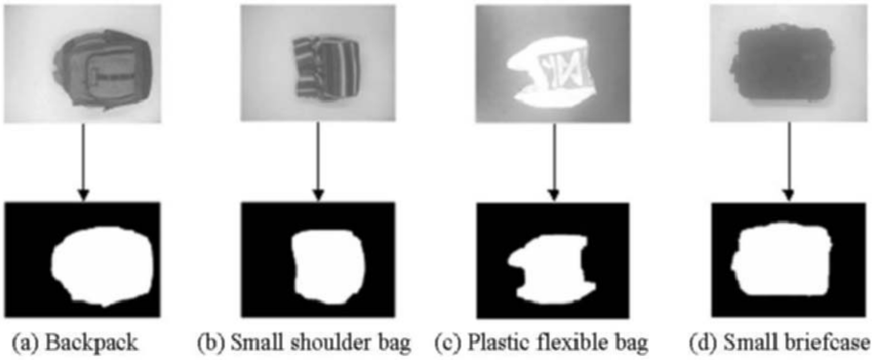


Fig. 3.1. Image processing operations

4 Experimental Results

4.1 Methodology

The heart of the support vector machine design is the kernel selection. Kernels project the data into a high dimensional feature space and thereby increase the computational power of the linear learning machines [16, 3]. The kernels chosen for the bag classification problem are the most common ones used in support sector machines, i.e., the polynomial and the Gaussian RBF kernels [3]. The kernels were tested using the K-fold cross validation procedure with three subsets on a training set that contained 80 bag images (20 of each of the four classes). For testing, 40 bag images (10 of each class) were used. The SVMs procedure was implemented in the "OSU SVM MATLAB Toolbox" [9].

Two experiments were performed. In the first one, described in Section 4.2, a kernel optimization procedure was conducted, using the nine bag features described in Section 3, for finding the optimal polynomial degrees and RBF sigmas. In the second experiment, described in Section 4.3, an optimal feature selection was conducted for finding the number and combination of features that results in the highest classification rate using kernel optimization.

4.2 Kernel Optimization Experiment for Bag Classification

Classification process was performed for the range of 1-100 degrees / sigmas applying both kernels for finding the optimal polynomial degrees and RBF sigmas using all the nine features mentioned in Section 3.

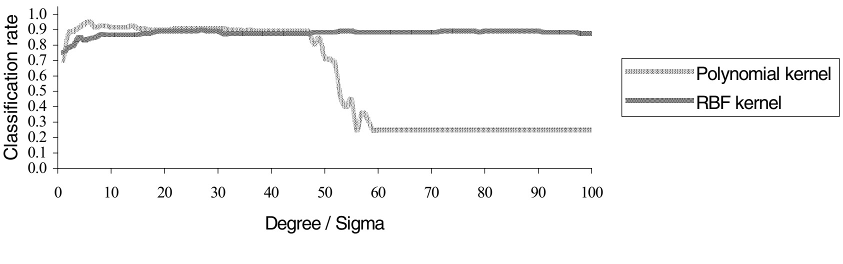
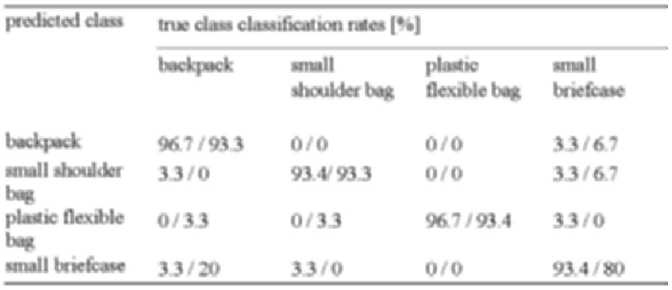


Fig. 4.2.1. Classification rate vs. degree and sigma for four bags classification using nine image features

As can be seen in Fig. 4.2.1 for the nine bag features case, the highest average classification rate between the three subsets achieved was 95% using a polynomial kernel with six degree and 90% using a RBF kernel with 27 sigma. The confusion matrices that summarize the results are shown in Table 4.2.1, where the upper and lower diagonal values in each cell correspond to percent of correct classifications for the polynomial and RBF kernels, respectively.

Table 4.2.1. Confusion matrices for polynomial/RBF kernels for a four bag classification using nine image features



predicted class	true class classification rates [%]			
	backpack	small shoulder bag	plastic flexible bag	small briefcase
backpack	96.7 / 93.3	0 / 0	0 / 0	3.3 / 6.7
small shoulder bag	3.3 / 0	93.4/ 93.3	0 / 0	3.3 / 6.7
plastic flexible bag	0 / 3.3	0 / 3.3	96.7 / 93.4	3.3 / 0
small briefcase	3.3 / 20	3.3 / 0	0 / 0	93.4 / 80

4.3 Optimal Feature Selection

A full enumeration feature selection procedure was performed to choose a set of optimal features to improve the results achieved in Section 4.2. Since there are up to nine features to be used in the classification procedure, there are 2^9 - 1 = 511 combinations for selection. The classification process was performed for the range of 1-100 degrees / sigmas applying both kernels to find the optimal set of bag image features corresponding to the optimal polynomial degrees and RBF sigmas giving the highest classification results. As can be seen from Fig 4.3.1, the largest average classification rate can be obtained by using only four features for both the polynomial and RBF kernels.

Details of the feature selections and the average classification rates appear in Table 4.3.1. The polynomial kernel's average classification rate of 96.25% was superior to that of the RBF (91.6%). It is noted that for the polynomial kernel, the minimum set of features (among the ties) consists of four features: bounding box ratio, major axis length, extent and roundness. Also, from Table 4.3.1 these features correspond to the optimal polynomial kernel of degree nine.

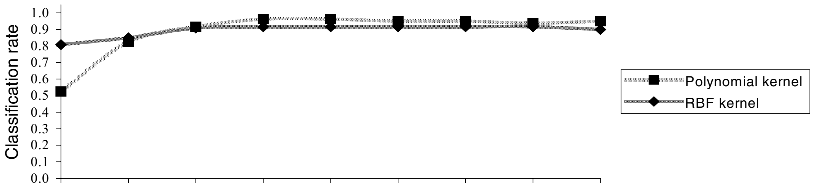
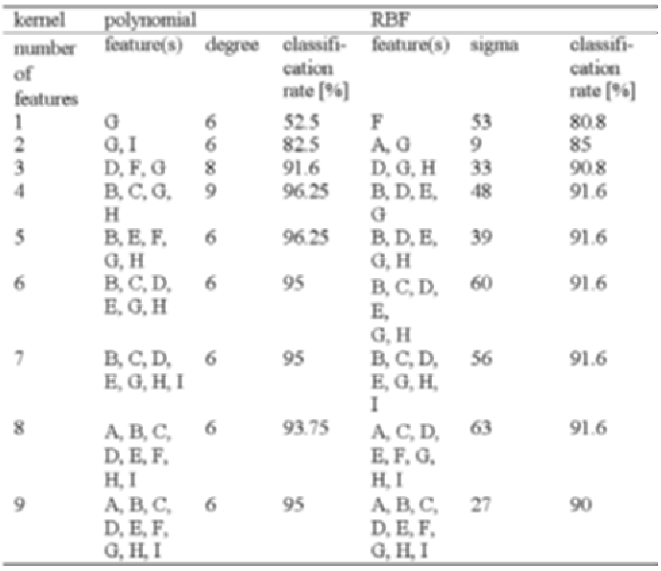


Fig. 4.3.1. Classification rate as a function of number of features using the optimal degrees / sigmas

Table 4.3.1. Feature selection results for the highest classification rates achieved



kernel	polynomial			RBF		
number of features	feature(s)	degree	classification rate [%]	feature(s)	sigma	classification rate [%]
1	G	6	52.5	F	53	80.8
2	G, I	6	82.5	A, G	9	85
3	D, F, G	8	91.6	D, G, H	33	90.8
4	B, C, G, H	9	96.25	B, D, E, G	48	91.6
5	B, E, F, G, H	6	96.25	B, D, E, G, H	39	91.6
6	B, C, D, E, G, H	6	95	B, C, D, E, G, H	60	91.6
7	B, C, D, E, G, H, I	6	95	B, C, D, E, G, H, I	56	91.6
8	A, B, C, D, E, F, H, I	6	93.75	A, C, D, E, F, G, H, I	63	91.6
9	A, B, C, D, E, F, G, H, I	6	95	A, B, C, D, E, F, G, H, I	27	90

A Area, B Bounding Box Ratio, C Major Axis Length, D Minor Axis Length, E Eccentricity, F Equivalent Diameter, G Extent, H Roundness, I Convex Perimeter.

Another view of the results are depicted in Fig. 4.3.2, using the optimal set of features (bounding box ratio, major axis length, extent and roundness).The curve shows that the average classification rate peaks for an optimal polynomial kernel degree nine.

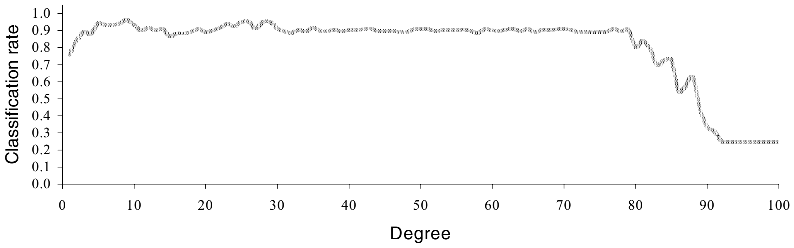
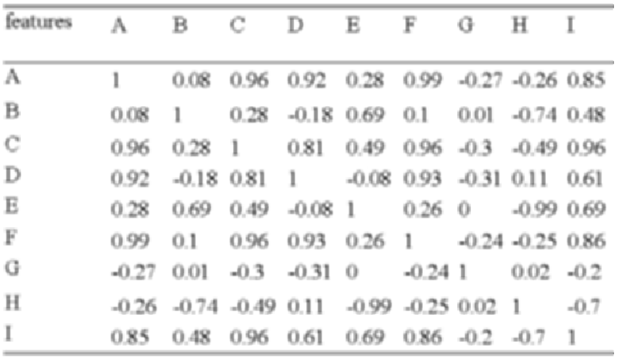


Fig. 4.3.2. Classification rate vs. degree for four-bag classification using optimal features (bounding box ratio, major axis length, extent and roundness)

5 Analysis of the Results

The polynomial kernel's classification rate of 96.25% was superior to that of the RBF (91.6%) using only four out of the nine original features. The small number of features deemed to be optimal was hypothesized to be due to correlation between the features. This is verified by examining the correlation matrix (Table 5.1), which exhibited correlation coefficients as high as 0.99.

Table 5.1. Correlation matrix

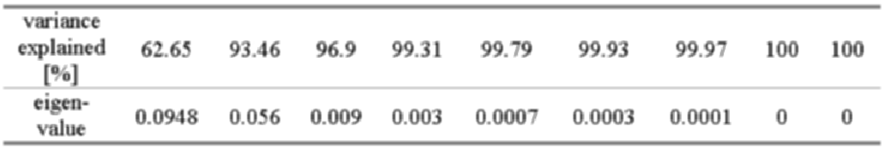


features	A	B	C	D	E	F	G	H	I
A	1	0.08	0.96	0.92	0.28	0.99	-0.27	-0.26	0.85
B	0.08	1	0.28	-0.18	0.69	0.1	0.01	-0.74	0.48
C	0.96	0.28	1	0.81	0.49	0.96	-0.3	-0.49	0.96
D	0.92	-0.18	0.81	1	-0.08	0.93	-0.31	0.11	0.61
E	0.28	0.69	0.49	-0.08	1	0.26	0	-0.99	0.69
F	0.99	0.1	0.96	0.93	0.26	1	-0.24	-0.25	0.86
G	-0.27	0.01	-0.3	-0.31	0	-0.24	1	0.02	-0.2
H	-0.26	-0.74	-0.49	0.11	-0.99	-0.25	0.02	1	-0.7
I	0.85	0.48	0.96	0.61	0.69	0.86	-0.2	-0.7	1

A Area, B Bounding Box Ratio, C Major Axis Length, D Minor Axis Length, E Eccentricity, F Equivalent Diameter, G Extent, H Roundness, I Convex Perimeter.

To further confirm the collinearity between the features we performed a principle component analysis (PCA), where a reduction to four components accounted for 99.3% of the variation for each of the bag types. Eigenvalues of the covariance matrix and the cumulative percentage of the total variance in the observations explained by the eigenvectors are shown in Table 5.2.

Table 5.2. Eigenvalues and variance explained matrix



variance explained [%]	62.65	93.46	96.9	99.31	99.79	99.93	99.97	100	100
eigen-value	0.0948	0.056	0.009	0.003	0.0007	0.0003	0.0001	0	0

It is noted, that the reduction from nine to four features was done by a complete enumeration feature selection method. It is a coincidence that a PCA allowed a reduction to four "components", but the PCA was not used to represent the feature reduction through a linear combination of the original features, as in the usual case.

6 Conclusions

Multi-category bag classification for four-bag classes was performed using SVMs. The SVMs procedure was tested using polynomial and RBF kernels. A K-fold cross validation procedure with three subsets was used. In a kernel optimization experiment using nine popular shape features, classification rates of 95% and 90% were achieved using a polynomial kernel of degree six and a RBF kernel with 27 sigma, respectively. The confusion matrices indicate that decreased classification rates may be due to the inability to discriminate between the backpacks and the small briefcases bag images. To improve these results, a full enumeration feature selection procedure for choosing a set of optimal features for describing a bag image was performed. The set of optimal features found was: bounding box ratio, major axis length, extent and roundness. Using these features a classification rate of 96.25% was obtained for a polynomial kernel of degree nine. It was also found that using more than four features resulted in no improvement. The resulting optimal reduction of features from nine to four was hypothesized to be due to correlation between the features. The collinearity between the features was confirmed using principle component analysis, where a reduction to four components accounted for 99.3% of the variation for each of the bag types. Future work includes a comparison of the SVM classifier with that of using a PCA classifier.

Acknowledgments

This project was partially supported by the Ministry of Defense MAFAT Grant No. 1102, and the Paul Ivanier Center for Robotics Research and Production Management, Ben-Gurion University of the Negev. The assistance of Prof. Ehud Menipaz is gratefully acknowledged.

References

[1] Bennett, K.P. and Bredensteiner, E.J. (1999), “Multicategory classification by support vector machines”, *Computational Optimizations and Applications*, vol. 12, pp. 53-79.

[2] Cortes, C. and Vapnik, V. (1995), “Support vector networks”, *Machine Learning*, vol. 20, pp. 1-25.

[3] Cristianini, N. and Shawe-Taylor, J. (2003), *Support Vector Machines and other Kernel-based Learning Methods*, Cambridge University Press, Cambridge, U.K.

[4] Guo, G.D., Li, S.Z., and Chan, K.L. (2000), “Learning similarity for texture image retrieval”, *Proceedings of the European Conference on Computer Vision*, pp. 178-190.

[5] Heisele, B., Ho, P., Wu, J., and Poggio, T. (2003), "Face recognition: component-based versus global approaches", *Elsevier Science Inc.*, New York, U.S.A., vol. 91, pp. 6-21.

[6] Hong, P., Tian, Q., and Huang, T.S. (2000), "Incorporate support vector machines to content-based image retrieval with relevance feedback", *Proceedings of the International Conference on Image Processing*, vol. 3, pp. 750-753.

[7] Kartoun, U. (2003), "A human-robot collaborative learning system using a virtual reality telerobotic interface", *Ph.D. Thesis Proposal, Department of Industrial Engineering and Management at the Ben-Gurion University of the Negev, Israel.*

[8] Kartoun, U., Stern, H., and Edan, Y. (2004), "Virtual reality telerobotic system", *e-ENGDET 2004 4th International Conference on e-Engineering and Digital Enterprise Technology*, Leeds Metropolitan University Yorkshire, U.K.

[9] Ma, J. and Ahalt, S. (2003), OSU SVM Classifier Matlab Toolbox (ver. 3.00).

[10] The MathWorks Inc., (1998), Matlab Software, Image Processing User's Guide.

[11] Nakajima, C., Pontil, M., Heisele, B., and Poggio, T. (2000), "Person recognition in image sequences: the MIT espresso machine system", *IEEE Transactions on Neural Networks.*

[12] Osuna, E., Freund, R., and Girosi, F. (1997), "Training support vector machines: an application to face detection", *Proceedings of Computer Vision and Pattern Recognition*, pp.130-136.

[13] Otsu, N. (1979), "A threshold selection method from gray level histograms", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, pp. 62-66.

[14] Schölkopf, B., Burges, C., and Vapnik, V. (1995), "Extracting support data for a given task", In U. Fayyad and R. Uthurusamy, editors. *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, AAAI Press, pp. 252-257.

[15] Shanahan, J.G. and Roma, N. (2003), "Boosting support vector machines for text classification through parameter-free threshold relaxation", *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, pp. 247-254.

[16] Vapnik, V. (1998), *Statistical Learning Theory*, JohnWiley and Sons, New York.

The Error Bar Estimation for the Soft Classification with Gaussian Process Models

J.B. Gao[1] * ** and Lei Zhang[2]

[1] School of Information Technology
Charles Sturt University, Bathurst, NSW 2795, Australia
jbgao@csu.edu.au

[2] School of Mathematics, Statistics and Computer Science
The University of New England, Armidale, NSW 2351, Australia
lzhang8@turing.une.edu.au

Summary. In this paper, we elaborate on the well-known relationship between Gaussian Processes (GP) and Support Vector Machines (SVM) under some convex assumptions for the SVM loss function for the binary classification. We also investigate the relationship between the Kernel method and Gaussian Processes. This paper concentrates mainly on the derivation of the error bar approximation for classification. An error bar formula is derived based on the convex loss function approximation.

1 Introduction

In the last decade neural networks have been used to tackle regression and classification problems, with some notable successes [1]. It has also been widely recognized that they form a part of a wide variety of nonlinear statistical techniques that can also be used for these tasks. In particular, Bayesian models which are based on Gaussian priors, both on the parameter spaces and function spaces are becoming increasingly popular in the neural computation community (see e.g. [9,10,22,23]). Several well established results in Reproducing Kernel Hilbert Spaces (RKHS) and Gaussian Processes are proving to be very useful in the application of support vector machine methods in classification and regression [20]. It has also been shown recently that Gaussian Processes and Support Vector Machines (SVM) are closely related, see eg. [21].

Both approaches are nonparametric, this means that they can allow for infinitely many parameters to be tuned, but increasing with the amount of

* This author is supported by the Natural Science Foundation of China (Grant No.: 60373090)

** Author to whom all the correspondences should be addressed.

data, only a finite number of the parameters are active. The models generated may be understood as generalizations of neural networks with single layer perceptrons. After the models construction, it is a fundamental task to estimate the prediction errors associated with these models. Error estimates (esp. error bars) are useful as they allow confidence intervals to be placed around model predictions for unseen data.

Recently, Penny et al. [12, 13] consider the error bar problem for both the linear and nonlinear neural network regression models under the Gaussian noise assumption. Their results are based on Geman's prediction error decomposition, i.e., model bias, model variance and target noise, [5].

We consider here the error bar for the soft classification problem which means that a Gaussian noise assumption is no longer valid. The main aim of this paper is to derive a formula for the error bar of soft classification [21] generalization. The approach taken is to employ the α-technique proposed in this paper, based on the structure of the estimation solution. The basic idea comes from the support vector machine (SVM) optimization algorithm.

In the following we will concentrate on the case where we have only two classes, and where the expected misclassification rate is used as the criterion. This is the case in which SVMs are best developed. In this situation the label y is either 1 or -1. A classification rule is a mapping from $\mathbb{R}^d$ to $\{1, -1\}$. It is easy to see that the expected misclassification rate R of any classification rule h can be written as

$$R = E[|Y - h(X)|/2] = E[1 - Yh(X)]_+/2.$$

In section 2 we review the basic concepts of SVM for classification problems and section 3 is dedicated to introducing some soft cost functions. In section 4 we utilize the α-technique to derive the error bar formula for the Gaussian process classification based on maximum a posterior estimation.

2 Support Vector Machine Method for Classification

Support Vector Machines (SVMs) have been the subject of intense research activity from within the neural networks community over the last few years; for tutorial introductions and overviews of recent developments see [14, 17]. In this section, we briefly review the use of SVM concepts in the (two-class) classification problems. Interested readers may consult [3, 4, 19] for details. Suppose we are given a set $\mathcal{D}$ of N training examples, each of the form $(\mathbf{x}_i, y_i)$ with a binary output $y_i = \pm 1$ indicating to which of the two possible classes the input $\mathbf{x}_i$ belongs. The inference task is to predict the correct label y on a new test point $\mathbf{x}$. In the Bayesian framework, this is done by using the posterior distribution of the actual (unknown) relation $h(\mathbf{x})$. There are two approaches to deal with this posterior distribution, i.e., in the parametric spaces (feature spaces) view and the function spaces view.

The basic SVM idea is to map the inputs $\mathbf{x}$ from the input space $\mathbb{R}^m$ onto vectors $\phi(\mathbf{x})$ in some high-dimensional feature space $\mathcal{F}$ with an inner-product $(\cdot,\ \cdot)$; ideally, in this feature space, the classification problem should be linearly separable. Consider the case when the data is linearly separable in $\mathcal{F}$. The SVM constructs a hyperplane $(\mathbf{w}, \phi(\mathbf{x})) + b$ in $\mathcal{F}$ for which the separation between the positive and negative examples is maximized. The separation means that the SVM hyperplane obeys $y_i((\mathbf{w}, \phi(\mathbf{x}_i)) + b) > 0$ for all index i. In practice, it is solved by specifying the margin and minimizing the length of vector $\mathbf{w}$ [17]. This finally leads to the optimization problem: Find a vector $\mathbf{w}$ in $\mathcal{F}$ and an offset b such that $\frac{1}{2}\|\mathbf{w}\|^2$ is minimized, subject to the constraint that $y_i((\mathbf{w}, \phi(\mathbf{x}_i)) + b) \geq 1$ for all i. This is a high-dimensional optimization problem which can be transferred into a quadratic programming (QP) problem by writing the vector $\mathbf{w} = \sum_{i=1}^N \alpha_i y_i \phi(\mathbf{x}_i)$. For those α_i's greater than zero, the corresponding training examples must lie along the margins of the decision boundary (by the Kuhn-Tucker theorem) and these examples are called the support vectors.

When the training examples are not separable in the feature space $\mathcal{F}$, the SVM algorithm introduces non-negative slack variables $\xi_i \geq 0$ $(i = 1, 2, \cdots, N)$. In order to control the amount of slack allowed, a penalty term must then be added to the objective function $\frac{1}{2}\|\mathbf{w}\|^2$. The most widely used choice is $\lambda \sum_{i=1}^N \xi_i$ with a penalty coefficient λ, and a more general penalty term$\lambda \sum_{i=1}^N c(\xi_i)$ can be considered with a convex cost function $c(\xi)$. The resultant problem is now

$$\text{minimize}\frac{1}{2}\|\mathbf{w}\|^2 + \lambda \sum_{i=1}^{N} c(\xi_i), \tag{2.1}$$

subject to $y_i((\mathbf{w}, \phi(\mathbf{x}_i)) + b) \geq 1 - \xi_i$, $i = 1, 2, \cdots, N$. Here λ is a user-defined regularization parameter controlling the trade-off between model complexity and training error, and ξ_i measures the difference between $(y_i(\mathbf{w}, \phi(\mathbf{x}_i)) + b)$ and 1.

The above framework for SVM can be interpreted from a probability viewpoint with Bayesian rule on the fact that (2.1) can be regarded as defining a (negative) log-posterior probability for the vector $\mathbf{w}$ and b given the observation data set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$. The first term in (2.1) describes the prior $P(\mathbf{w}) \sim \exp(-\frac{1}{2}\|\mathbf{w}\|^2)$ for the vector $\mathbf{w}$, i.e., the normalized Gaussian distribution on the feature space $\mathcal{F}$. If we have no prior about the offset b, then the distribution of b can be assumed to be flat. The second term in (2.1) can be identified as the log-likelihood of the vector $\mathbf{w}$. Hence optimizing (2.1) can be regarded as finding the *maximum a posterior* (MAP) estimate $\mathbf{w}_{\text{MP}}$ of $\mathbf{w}$ [18].

Because only the latent function value $a(\mathbf{x}) = (\mathbf{w}, \phi(\mathbf{x}_i))$ is related to the slack variable ξ_i in the second term of (2.1), rather than on $\mathbf{w}$ and b individually, it makes sense to express the prior directly as a distribution over function $a(\mathbf{x})$. In this case, the function values $a(\mathbf{x})$'s on the distinct point $\mathbf{x}$ have a joint Gaussian distribution because $\mathbf{w}$ is a Gaussian field on the feature space $\mathcal{F}$. Then we can specify a Gaussian process over the function $a(\mathbf{x})$, with a covariance function $C(\mathbf{x}, \mathbf{x}')$, thus the intermediate vector $\mathbf{w}$ is condensed in a function. This generalization from the conventional SVMs to GPs has been recently noted by a number of authors, e.g. [7, 15, 21, 22]. Finally, the GP framework for the classification problem is then given by minimizing the negative log-posterior as follows:

$$\text{minimize } \frac{1}{2}\sum_{i,j=1}^{N} a(\mathbf{x}_i)C^{-1}(\mathbf{x}_i, \mathbf{x}_j)a(\mathbf{x}_j) + \lambda\sum_{i=1}^{N} c(\xi_i) \tag{2.2}$$

subject to $y_i a(\mathbf{x}_i) \geq 1 - \xi_i$ for all $i = 1, 2, \cdots, N$.

As in the standard SVM algorithm we want to find the minimum $a(\mathbf{x})$ of (2.2) which can be written in the form

$$a(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i C(\mathbf{x}_i, \mathbf{x}) + b \tag{2.3}$$

with a bias constant b.

The equations (2.1) and (2.2) represent the parameter space view and the function space view, respectively.

3 From Hard Classification to Soft Classification

The other important choice is the so-called convex cost function $c(\xi)$ employed in equations (2.1) and (2.2). There are many different choices for the function $c(\xi)$, each of which corresponds to a special SVM algorithm. The most widely used is the function $c(\xi) = [\xi]_+ = \xi I_{\{\xi>0\}}$ with the relationships $\xi_i = 1 - y_i((\mathbf{w}, \phi(\mathbf{x}_i) + b)$ and $\xi_i = 1 - y_i a(\mathbf{x}_i)$ in (2.1) and (2.2), respectively.

In general, the second term in (2.1) or (2.2) can be regarded as the measure of the misclassifications on the training set, while the first term can be used to control the complexity of the estimated model function. Optimizing (2.1) or (2.2) means putting on the penalty on the misclassification amount/rate. A lot of cost function can be found in [6, 16].

An approach can be taken to penalize the number of misclassifications on the given training set. This derives the so-called hard classification criteria which are recently discussed in [21] and [11]. The basic idea is to define the cost function $c(\xi) = [\xi]_*$ where $[\xi]_* = 1$ if $\xi > 0$ otherwise $[\xi]_* = 0$, or more general, $c(\xi) = [\xi - \epsilon]_*$ for some fixed constant $\epsilon < 1$. In this case, we should estimate a function $a(\mathbf{x})$ to minimize

$$\frac{1}{2}\sum_{i,j=1}^{N} a(\mathbf{x}_i)C^{-1}(\mathbf{x}_i, \mathbf{x}_j)a(\mathbf{x}_j) + \lambda\sum_{i=1}^{N}[1 - \epsilon - y_i a(\mathbf{x}_i)]_* \tag{3.1}$$

thereby penalizing the misclassification rate rather than the log likelihood.

Unfortunately the use of $[1 - \epsilon - y_i a(\mathbf{x}_i)]_*$ in (3.1) results in a nonconvex programming problem. In order to overcome this problem Bradley et al [2] consider replacing $[1 - \epsilon - y_i a(\mathbf{x}_i)]_*$ with other more tractable functions including the sigmoidal approximation etc. This approach has also been taken in [8] to get the moderated output of the SVM for the pattern $\mathbf{x}$.

Let us consider the most widely used cost function $c(\xi) = [\xi]_+$. This function is continuously differentiable except for at one point, and can be well approximated by the smooth function $\ln(1 + \exp(\xi))$. If we take $c(\xi) = \ln(1 + \exp(\xi))$, then the minimization problems (2.1) and (2.2) can be interpreted by the so-called soft classification concept by which one can penalize the relative risks on the training data set belonging to two classes, see [21]. In [8] the author also takes a similar approximation trick in which the cost function $c(\xi) = [\xi]_+$ was replaced with $c(\xi) = \sigma(\xi)\xi$ where $\sigma(\xi)$ is the sigmoid function $1/(1 + \exp(-\mu\xi))$ for some $\mu > 0$. This is because we can represent $[\xi]_+ = [\xi]_*\xi$ and $[\xi]_*$ can be approximated by a sigmoid function. But in this later case, we have no intuitive interpretation for the cost function. However all of these can be regarded as the (negative) log likelihood of the training data with a specifying noise process between the target values and the estimated values.

In the next section, we mainly focus on the cost function $c(\xi) = \ln(1 + \exp(\xi))$.

4 Computation Method for the Error Bar

The goal of this section is to derive the generalization error of the minimizer of (2.1) or (2.2) under the assumption for the cost function $c(\xi) = \ln(1+\exp(\xi))$. For our convenience, we re-write the equation (2.2) as follows

$$\text{minimize } \frac{1}{2}\sum_{i,j=1}^{N} a(\mathbf{x}_i)C^{-1}(\mathbf{x}_i, \mathbf{x}_j)a(\mathbf{x}_j) + \lambda \sum_{i=1}^{N} \ln(1 + \exp(\xi_i)) \tag{4.1}$$

where the minimization is taken over a function space of $a(\mathbf{x})$ and $\xi_i = 1 - y_i a(\mathbf{x}_i)$.

The minimization problem (4.1) can be linked with the problem of maximum a posterior estimate under the probabilistic interpretation, see [18], in which the first term in (4.1) can be regarded as the negative log prior of a Gaussian Process $a(\mathbf{x})$ and the second term is the negative likelihood of the training data set. By the Bayesian Rule, the solution of the problem (4.1) is the solution $a_{MP}(\mathbf{x})$ of MAP estimate under the above probabilistic framework.

Consider the solution of (4.1) in the function space generated by the function form (2.3). Thus (4.1) can be solved by an optimal procedure on the parameter vector $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_N)^T$. Let us write the solution of (4.1) in the form

$$a_{MP}(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i^{MP} y_i C(\mathbf{x}_i, \mathbf{x}) + b. \tag{4.2}$$

where $\alpha^{MP} = (\alpha_1^{MP}, \alpha_2^{MP}, \cdots, \alpha_N^{MP})^T$ is the optimal parameter determined by the optimization problem (4.1).

Under the assumption (2.3), the problem (4.1) is equivalent to a nonlinear convex optimization with respect to variables $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_N)^T$. For the sake of convenience, denote $\alpha_y = (\alpha_1 y_1, \alpha_2 y_2, \cdots, \alpha_N y_N)^T$. Let us take a second order approximation to (4.1) with respect to α_y, i.e., a Gaussian approximation to the posterior distribution of function $a(\mathbf{x})$ as shown in [18].

Denote

$$E(a(\mathbf{x})) = \frac{1}{2} \sum_{i,j=1}^{N} a(\mathbf{x}_i) C^{-1}(\mathbf{x}_i, \mathbf{x}_j) a(\mathbf{x}_j) + \lambda \sum_{i=1}^{N} \ln(1 + \exp(\xi_i)) \tag{4.3}$$

Substituting (2.3) into (4.3) results in the following nonlinear function

$$E(\alpha_y) = \frac{1}{2} \alpha_y^T \mathbf{C} \alpha_y + \lambda \sum_{i=1}^{N} \ln(1 + \exp(1 - y_i \alpha_y^T C(i, :))) \tag{4.4}$$

where $\mathbf{C} = [C(\mathbf{x}_i, \mathbf{x}_j)]$ is the covariance matrix corresponding to Kernel function $C(\mathbf{x}, \mathbf{x}')$, and $\mathbf{C}(i, :)$ is the i-th column of $\mathbf{C}$.

Here (4.4) can be viewed as the (negative) log-posterior distribution of the new parameter α_y as mentioned above. We assume that the posterior distribution can be approximated by a single Gaussian at the MAP solution α_y^{MP}. That means we can take the second order Taylor's expansion of $E(\alpha_y)$ at α_y^{MP} to approximate $E(\alpha_y)$, i.e.,

$$E(\alpha_y) \approx E(\alpha_y^{MP}) + \delta(\alpha_y)^T \mathbf{A} \delta(\alpha_y)$$

where $\delta(\alpha_y) = \alpha_y - \alpha_y^{MP}$ and $\mathbf{A} = \nabla^2_{\alpha_y}(E(\alpha_y))$ is the Hessian of $E(\alpha_y)$ at α_y^{MP}.

Thereby the vector $\delta(\alpha_y)$ is a joint Gaussian with mean $\mathbf{0}$ and covariance A^{-1}. Denote by $C(\mathbf{x}) = [C(\mathbf{x}, \mathbf{x}_1), \cdots, C(\mathbf{x}, \mathbf{x}_N)]^T$, then $a(\mathbf{x}) = \alpha_y^T \cdot C(\mathbf{x}) + b$. Thus we can write

$$a(\mathbf{x}) - a_{MP}(\mathbf{x}) = \delta(\alpha_y)^T \cdot C(\mathbf{x}).$$

Consequently, the posterior distribution of $a(\mathbf{x})$ will also be Gaussian with mean $a_{MP}(\mathbf{x}) = \sum\limits_{i=1}^{N} \alpha_i^{MP} y_i C(\mathbf{x}, \mathbf{x}_i) + b$ and variance

$$\sigma^2(\mathbf{x}) = C(\mathbf{x})^T \mathbf{A}^{-1} C(\mathbf{x}) \tag{4.5}$$

The formula (4.5) is the approximated error bar for the estimate $a_{MP}(\mathbf{x})$ under a single Gaussian approximation. To compute $\sigma^2(\mathbf{x})$ in (4.5), we have to determine the Hessian $\mathbf{A}$. First, define functions $r(\xi) = \frac{\exp(\xi)}{(1+\exp(\xi))^2}$ and $r_{1/2}(\xi) = \sqrt{r(\xi)} = \frac{\sqrt{\exp(\xi)}}{1+\exp(\xi)}$. By a direct computation, we can get the following relation:

$$\mathbf{A} = \mathbf{C} + \lambda\widetilde{\mathbf{C}}\widetilde{\mathbf{C}}^T \tag{4.6}$$

where $\widetilde{\mathbf{C}}$ is the matrix whose i-th column is the product of $r_{1/2}(\xi_i)$ with the i-th column of $\mathbf{C}$. Since A is a positive definite matrix, let us consider Schur's decomposition

$$\mathbf{A} = \mathbf{P}\Sigma\mathbf{P}^T$$

where $\Sigma = \text{diag}[\sigma_i]$ is a diagonal matrix consisted of eigenvalues of $\mathbf{A}$ and $\mathbf{P} = [\mathbf{v}_1, \cdots, \mathbf{v}_N]$ is a unitary matrix whose columns $\mathbf{v}_i$ is the eigenvectors of $\mathbf{A}$. Utilizing the fact that $\mathbf{P}^{-1} = \mathbf{P}^T$, $\sigma^2(\mathbf{x})$ in (4.5) can then be computed as

$$\sigma^2(\mathbf{x}) = C(\mathbf{x})^T\mathbf{A}^{-1}C(\mathbf{x}) = (\mathbf{P}^TC(\mathbf{x}))^T\Sigma(\mathbf{P}^TC(\mathbf{x})) = \sum_{i=1}^{N}\frac{1}{\sigma_i}\left(\mathbf{v}_i^T\cdot C(\mathbf{x})\right)^2$$

In this section, we have proposed a method for computing the error bar of soft classification problem with Gaussian Process. Our approach is based on the solution form (2.3) and all of induction focus on the intermediate parameter vector α_y (or α). Hence the basic derivation and results look like that in the case of generalized linear model estimation.

5 Conclusion

In this paper, we derive the error bar of the soft classification regularization based on the Gaussian process framework. We have focused on classification problems. Extension to regression problems for the calculation of error bars is also straight-forward. These will be investigated in the future. Although nonparametric Bayesian models which are based on Gaussian process priors on function spaces are infinite dimensional, the desired model can still assume a linear form of the covariance function at the training data set which results in the α technique used in this paper. Finally, as this work is based on MAP, variability in the hyperparameter (i.e. the regularization parameter λ) is ignored. This issue will be studied and the application of other Bayesian techniques, like Markov Chain Monte Carlo methods, will also be considered.

References

1. C.M. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, 1995.

2. P.S. Bradley and O. Mangasarian. Massive data discrimination via linear support vector machines. Mathematical Programming Technical Report 98-05, University of Wisconsin, Madison, 1998.
3. C.J.C. Burges. The geometry of support vector machines. *Neural Computation*, page submitted, 1998.
4. V. Cherkassky and F. Mulier. *Learning from Data – Concepts, Theory and Methods*. Wiley, New York, 1998.
5. S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4:1–58, 1992.
6. S.R. Gunn. Support vector machines for classification and regression. Technical report, ISIS, Department of Electronics and Computer Science, University of Southampton, 1998.
7. T.S. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *Proceedings of The Seventh International Workshop on Artificial Intelligence and Statistics*, 1998.
8. J. T.-Y. Kwok. Moderating the outputs of support vector machine classifiers. *IEEE-NN*, 10(5):1018, September 1999.
9. D.J. MacKay. Gaussian processes, A replacement for neural networks. NIPS tutorial 1997, Cambridge University, 1997.
10. R. Neal. *Bayesian Learning for Neural Networks*. Lecture Notes in Statistics. Springer, New York, 1996.
11. M. Opper and O. Winther. Gaussian process classification and SVM: Mean field results and leave-one-out estimator. Research report, Neural Computing Research Group, Aston University, Birmingham, UK, 1999.
12. W.D. Penny and S.J. Roberts. Neural network predictions with error bars. Technical report, Neural Systems Researhc Group, Imperial College of Science, Technology and Medicine, UK, 1997.
13. W.D. Penny and S.J. Roberts. Error bars for linear and nonlinear neural network regression models. Technical report, Neural Systems Researhc Group, Imperial College of Science, Technology and Medicine, UK, 1998.
14. B. Scholkopf, C. Burges, and A.J. Smola. *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA, 1999.
15. A. Smola, B. Schölkopf, and K.R. Müller. General cost functions for support vector regression. In T. Downs, M. Frean, and M. Gallagher, editors, *Proceedings of the Ninth Australian Conference on Neural Networks*, pages 79–83, Brisbane, Australia, 1998. University of Queensland.
16. A.J. Smola. *Learning with Kernels*. PhD thesis, Technischen Universität Berlin, Berlin, Germany, 1998.
17. A.J. Smola and B. Schölkopf. A tutorial on support vector regression. Technical Report TR-1998-030, Neuro COLT, Berlin, 1998.
18. P. Sollich. Probabilistic interpretations and Bayesian methods for support vector machines. Technical report, King's College London, London, UK, 1999.
19. V.N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.
20. G. Wahba. *Splines Models for Observational Data*, volume 59 of *Series in Applied Mathematics*. SIAM Press, Philadelphia, 1990.
21. G. Wahba. Support vector machines, reproducing kernel hilbert spaces, and randomized gacv. In B. Schölkopf, C.J.C. Burges, and A. Smola, editors, *Adcances in Kernel Methods: Support Vector Learning*, pages 69–88. MIT Press, Cambridge, Massachusetts, 1999.

22. C.K. Williams. Computing with infinite networks. In M.C. Mozer, M.I. Jordan, and T. Petsche, editors, *Neural Information Processing Systems*, volume 9, pages 295–301. MIT Press, 1997.
23. C.K. Williams. Prediction with gaussian processes: from linear regression to linear prediction and beyond. In M.I. Jordan, editor, *Learning in Graphical Models*, pages 599–621. MIT Press, Cambridge, Massachusetts, 1998.

Research of Mapped Least Squares SVM Optimal Configuration

Sheng ZHENG[1, 2], Jian LIU[2] and Jin-wen Tian[2]

1. China Three Gorges University, Yichang 443002, China
2. IPRAI, Huazhong Univ Sci and Tech, Wuhan 430074, China

Abstract. To determine the optimal configuration of the kernel parameters, the physical characteristic of the mapped least squares (LS) support vector machine (SVM) is investigated by analyzing the frequency responses of the filters deduced from the support value and LS-SVM itself. This analysis of the mapped LS-SVM with Gaussian kernel illustrates that the optimal configuration of the kernel parameter exists and the regulation constant is directly determined by the frequency content of the image. The image regression estimation experiments demonstrate the effectiveness of the presented method.

1 Introduction

In recent years, the support vector machine (SVM), based on statistical learning theory, as a powerful new tool for data classification and function estimation, has been developed (Vapnik 1995). Originally, SVMs are developed for pattern recognition problems. And now, with the introduction of -insensitive loss function, SVMs have been extended to solve nonlinear regression estimation and time-series prediction (Vapnik 1995). Especially, Suykens et al. (1999) proposed a modified version of SVM called least squares SVM (LS-SVM), which resulted in a set of linear equations instead of a quadratic programming problem, and extended the application of the SVM to online application. The introduction of mapping technique extends in further the application of the SVM to image processing application (Zheng et al. 2004).

Selection of kernel functions is an important researching branch in the area of SVMs. The choice of the kernel has a crucial effect on the performance. There are three typical ways for model selection with different criteria, each of which is a prediction of the generalization error, including Bayesian evidence framework (Tipping 2001), PAC (Smola 1998) and cross validation (Martin and Hirschberg 1996). In fact, support vector machines comprise a class of learning algorithms originally introduced for pattern recognition problems. Although their development was motivated by the results of statistical learning theory, the known bounds on their generalization performance are not fully satisfactory. In particular, the influence of the chosen kernel is far from being completely understood. Since the resulting function classes represented by the classifier are very large there always exists the risk of overfitting (Steinwart 2001). Given a set of sample data set, to determine the kernel type and the kernel parameters during the training process demands excessive attentions in order to achieve desired level of generalization.

At present, the Gaussian kernel is the most common one due to its good features (Broomhead and Lowe 1988). In the Gaussian kernel case, because the spread parameter σ is closely associated with generalization performance of SVM, how to choose an appropriate σ is worth pursuing. This paper is to discuss the issue of kernel model selection and parameters specification. The rest of paper is organized as follows. In section 2, the kernel parameter selection and the mapped LS-SVM are reviewed. In section 3 and 4, the physical characteristic of the mapped LS-SVM with Gaussian kernel and the link between the kernel parameters and the frequency content of the image are discussed. Finally, conclusion is given.

2 Mapped LS-SVM and Model Selection

2.1 Least Squares (LS) SVM

In the LS-SVM, the Vapnik's standard SVM classifier has been modified into the following LS-SVM formulation:

$$\min_{\boldsymbol{w},b,\boldsymbol{e}} J(\boldsymbol{w},b,\boldsymbol{e}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \gamma\frac{1}{2}\sum_{k=1}^{N} e_k^2 \tag{1}$$

subject to $y_k\left\{\boldsymbol{w}^T\phi(\boldsymbol{x}_k)+b\right\}=1-e_k, k=1,\cdots,N$, defining the Lagrangian function

$$L(\boldsymbol{w},b,\boldsymbol{e};\boldsymbol{\alpha}) = J(\boldsymbol{w},b,\boldsymbol{e}) - \sum_{k=1}^{N}\alpha_k\left\{y_k\left[\boldsymbol{w}^T\phi(\boldsymbol{x}_k)+b\right]-1+e_k\right\} \tag{2}$$

where the α_k is the Lagrange multipliers. The optimality condition leads to the following (*N*+1)×(*N*+1) linear system

$$\begin{bmatrix} \boldsymbol{0} & \boldsymbol{1}^T \\ \boldsymbol{1} & \boldsymbol{Z}+\gamma^{-1}\boldsymbol{I} \end{bmatrix}\begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{Y} \end{bmatrix} \tag{3}$$

With

$\boldsymbol{Y}=\left[y_1,\cdots,y_N\right]^T, \boldsymbol{1}=\left[1,\cdots,1\right]^T, \boldsymbol{\alpha}=\left[\alpha_1,\cdots,\alpha_N\right]^T, \boldsymbol{Z}_{k,l}=\phi(\boldsymbol{x}_k)^T\phi(\boldsymbol{x}_l)=K(\boldsymbol{x}_k,\boldsymbol{x}_l)$.

Then the resulting LS-SVM model for function estimation becomes:

$$y(\boldsymbol{x}) = \sum_{k=1}^{N}\alpha_k K(\boldsymbol{x},\boldsymbol{x}_k)+b \tag{4}$$

where parameters α_k and *b* are based on the solution to Eq. (3).

2.2 Model Selection

Some machine learning methods focus on the problem of approximating a target function *f*: $\boldsymbol{X}\rightarrow\boldsymbol{Y}$ using the information from a sample of examples ($\boldsymbol{x}_i$, y_i), for *i*=1, …, *N*, where $\boldsymbol{x}\in\boldsymbol{X}$ and $y_i=f(\boldsymbol{x}_i)$. The learner (a learning algorithm) builds a hypothesis function $h\in H$ that approximates the target function *f* by determining the parameter setting θ of the hypothesis. Most of times, however, the hypothesis has certain parameters, which the learner itself is unable to determine.

Thus, for a particular learner *L*, there are two kinds of parameters in a hypothesis: the parameters τ (training parameters) that are determined automatically by *L* and the parameters λ (adjustable parameters) that are not determined by L. The adjustable parameters are typically determined by subjective judgment human, which bases on previous experience, rule of thumbs or heuristics provided by authors and practitioners of the learning algorithm. More precisely, the adjustable parameters are determined by minimizing an estimative of the true prediction error over a set of adjustable parameters that is known to work well on the training dataset. During the training process for SVMs the kernel parameters are the adjustable parameters and need to be specified to minimize the generalization error (true prediction error). For example, we have to determine the kernel parameters (one (RBF) or more (e.g. polynomial kernel)) and the regularization constant C in the standard SVM or the γ in the LS-SVM. To find best kernel parameters, three typical ways are Bayesian evidence framework (Tipping 2001), PAC (Smola 1998) and Cross validation (Martin and Hirschberg 1996).

Among the three approaches, the most frequently used method is the third but the problem is that the computational cost is the highest, because the learning problem must be solved k times. Other techniques, including Genetic algorithm, Fisher discrimination, etc. are also introduced. All these methods are to predict the generalization error by many times tries or iteratively solving the SVMs. Few directly uses the property of the dataset to determine the model and kernel parameters. For a practical problem, the sufficient and well-defined dataset are the base and their distribution property is the key to select the kernel parameters. To establish the link between the trained dataset and the kernel parameters, for the sake of convenience, the physical property of a special SVM, the mapped LS-SVM, and the role of the kernel parameters are investigated.

2.3 Mapped LS-SVM

As we know, solving a LS-SVM involves inverting an $N \times N$ square matrix, where N is the number of data points. This can be a formidable problem when N is large. Even in the small dataset, the repeatedly computation of matrix inverse is a very heavy burden for a real time system.

In the image processing, for example, image compression, the input image is usually split up into blocks or vectors of 8×8, or 16×16 pixels. Obviously, the input vector is the pixel coordinate (r, c) and the output is the intensity value. For an image block, the input points are usually of the form $\{(r + \delta r, c + \delta c): |\delta r| \le m, |\delta c| \le n\}$ and all such set of points can be translated to the same set $\{(\delta r, \delta c): |\delta r| \le m, |\delta c| \le n\}$ by subtracting (r, c) from all the vectors, where m and n are respectively the half size of the image block's horizontal and vertical pixels. By adding (r, c) to the set $\{(\delta r, \delta c): |\delta r| \le m, |\delta c| \le n\}$, the original image forms are reconstructed. With this mapping technique, the LS-SVM learning problem for image processing can be transformed into the problem with the same set of input vectors but different sets of labels. For a specified image block size, the set $\{(\delta r, \delta c): |\delta r| \le m, |\delta c| \le n\}$ is a constant vector space.

For Eq. (3), it is in fact only necessary to invert the N×N matrix

$$\boldsymbol{\Omega} = K(\boldsymbol{x}_i, \boldsymbol{x}_j) + \gamma^{-1}\boldsymbol{I} \tag{5}$$

Together with the 2nd and the 1st rows of Eq. (3) gives the explicit solution

$$b = \frac{\boldsymbol{1}^T\boldsymbol{\Omega}^{-1}\boldsymbol{Y}}{\boldsymbol{1}^T\boldsymbol{\Omega}^{-1}\boldsymbol{1}}, \boldsymbol{\alpha} = \boldsymbol{\Omega}^{-1}(\boldsymbol{Y} - b\boldsymbol{1}) \tag{6}$$

In the defined constant vector space, all the LS-SVM parameters and the input vector can be unchanged, and the $\boldsymbol{\Omega}$ can be constant. In this way, matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ can be used as constants and be pre-calculated by

$$\boldsymbol{A} = \boldsymbol{\Omega}^{-1}, \boldsymbol{B} = \frac{\boldsymbol{1}^T\boldsymbol{\Omega}^{-1}}{\boldsymbol{1}^T\boldsymbol{\Omega}^{-1}\boldsymbol{1}} \tag{7}$$

then the solution of Eq. (6) can be given by

$$b = \boldsymbol{BY}, \boldsymbol{\alpha} = \boldsymbol{A}(\boldsymbol{Y} - b\boldsymbol{1}) \tag{8}$$

These matrixes $\boldsymbol{A}$ and $\boldsymbol{B}$, however, depend only on the input vectors ($\boldsymbol{x}_k$), but not on their labels (y_k). Thus, in the image processing application, the learning computational complexity of the LS-SVM can be reduced to only about $O(N^2)$, just only a matrix multiplication, where the N=(2m+1)(2n+1). Compared with the standard LS-SVM, the mapped LS-SVM is very simple. However, for a given image, how to predetermine the kernel parameters and the regulation constant, say the $\boldsymbol{\Omega}$, is still a problem.

3 Physical Property of Mapped LS-SVM

To predetermine kernel parameters and the regulation constant, say the $\boldsymbol{\Omega}$, it is important to understand the physical property of the mapped LS-SVM and the role of the kernel parameters and the regulation constant.

3.1 Image Intensity Surface Function

Let R and C be the index sets of an image neighborhood that satisfy the symmetric conditions. For example, let R be defined as R={–1 0 1} and C be defined as C={–1 0 1}. In the LS-SVM with the Gaussian kernel, the fitted intensity surface function over the defined vector space R×C, can be rewritten as $g(r,c)=\sum_{k=1}^{N}\alpha_k \exp\left\{-\left(\left|r-r_k\right|^2+\left|c-c_k\right|^2\right)/\sigma^2\right\}+b$, where ($r$, c) is the coordinate of the pixel, (r_k, c_k) is the coordinate of the observed pixel, and the g(r, c) is the approximated intensity value, b and α_k are based on the mapped LS-SVM solution in Eq. (8).

3.2 Filters of Mapped LS-SVM

The support value α_k of the observed image pixel g(r_k, c_k) can be easily calculated using Eq. (8). The Eq. (8) can be rewritten as follows:

$$b=\boldsymbol{BY}, \boldsymbol{\alpha}=\boldsymbol{A}(\boldsymbol{I}-\boldsymbol{1B})\boldsymbol{Y}=\boldsymbol{O}_{\alpha}\boldsymbol{Y} \tag{9}$$

where $\boldsymbol{O}_{\boldsymbol{\alpha}}$ is a N×N matrix defined by $\boldsymbol{A}(\boldsymbol{I}\text{-}\boldsymbol{1B})$. Notice that, support value α_k of the observed image pixel g(r_k, c_k) over R×C is determined by the multiplication of the k^{th} row vector of the matrix $\boldsymbol{O}_{\boldsymbol{\alpha}}$ and the vector $\boldsymbol{Y}$. That is, the corresponding support value can be computed individually as a linear combination of the measured intensity values g(r_k, c_k), k=1,…,N. The weight associated with each g(r_k, c_k) is respectively determined by the corresponding row vector of the matrix $\boldsymbol{O}_{\boldsymbol{\alpha}}$. For a rectangular neighborhood, reshape the i^{th} row vector of the matrix $\boldsymbol{O}_{\boldsymbol{\alpha}}$ and the vector $\boldsymbol{Y}$, then the weight kernels become the support value operator $\boldsymbol{L}_{\alpha i}$, i=1,…, N.

With the Eq. (9), the fitted intensity surface function over the constant vector space, $R\times C$, can be rewritten as follows:

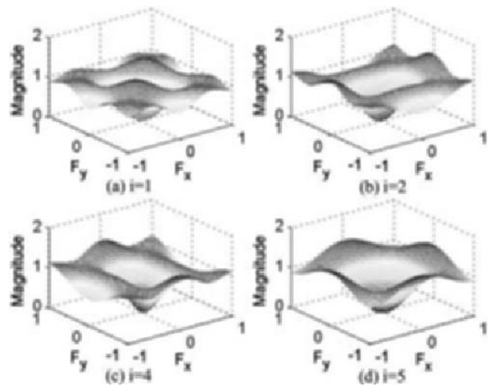
$$g(r,c) = \boldsymbol{F\alpha} + b = \boldsymbol{FA}(\boldsymbol{I} - \boldsymbol{1})\boldsymbol{BY} + \boldsymbol{BY} = \boldsymbol{F}_h\boldsymbol{Y} + \boldsymbol{F}_l\boldsymbol{Y} \tag{10}$$

where $\boldsymbol{F} = [f_1, \cdots, f_N], f_k = \exp\left\{-\left(\left|r - r_k\right|^2 + \left|c - c_k\right|^2\right)/\sigma^2\right\}, k = 1, \cdots, N$, $\boldsymbol{F}_l$=$\boldsymbol{B}$, and $\boldsymbol{F}_h$=$\boldsymbol{FA}(\boldsymbol{I}$-$\boldsymbol{1B})$.Notice that, at the defined neighborhood center (0, 0), where r=0 and c=0, all the $\boldsymbol{F}_h$ and $\boldsymbol{F}_l$ are determined by the predefined $\boldsymbol{\Omega}$. Therefore, the $\boldsymbol{F}_h$ and $\boldsymbol{F}_l$ can be pre-calculated. Eq. (10) shows that, the fitted intensity value with the mapped LS-SVM can be computed individually as a linear combination of the observed intensity values g(r_k, c_k). The weight associated with each g(r_k, c_k) is respectively determined by $\boldsymbol{F}_h$ and $\boldsymbol{F}_l$. For a rectangular neighborhood, we reshape the $\boldsymbol{F}_h$ and $\boldsymbol{F}_l$, and the weight kernels become the new image filters $\boldsymbol{L}_h$ and $\boldsymbol{L}_l$. Then the fitted intensity value is decomposed of two parts, which can be computed independently by convolving the image with the corresponding operators. For the neighborhood center (0, 0), the $\boldsymbol{L}_h$ and $\boldsymbol{L}_l$ are respectively determined by the vector $\boldsymbol{F}_h$ and $\boldsymbol{F}_l$ when r=0 and c=0, which correspond to the low- and high- pass filters respectively.

3.3 Physical Interpretation

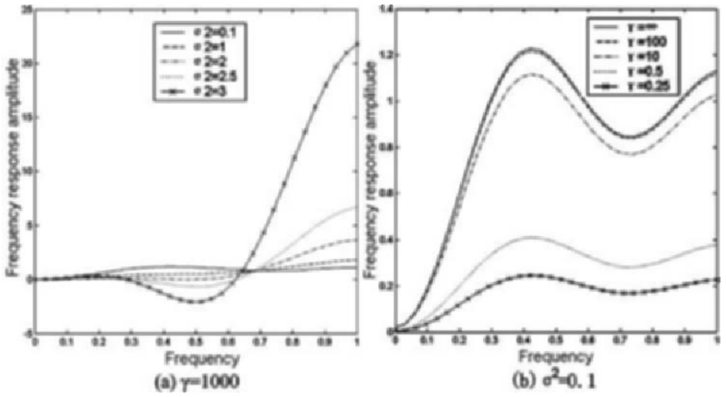
The calculation of the support values and approximation of the underlying gray level intensity surface are translated into the convolution of image with the corresponding operators. With the deduced operators, the physical characteristic of the mapped LS-SVM and the role of the kernel parameters can be investigated by analyzing their frequency responses.

There are nine support value operators $\boldsymbol{L}_{\alpha i}$ (i=1, …, 9) over the neighborhood (3×3) deduced from the mapped LS-SVM with Gaussian kernel. Figure 1 shows the frequency responses of the several corresponding support value operators, where the σ^2 is 0.1 and the γ is 1000. Obviously, the support value operator $\boldsymbol{L_\alpha}$ is in fact a set of high-pass filters. Except for the neighborhood center (0,0), all the filters are asymmetric. According to these filters, the support values correspond to the high-frequency content. As the features at the finest scale and the edge are at the high frequencies, the pixels at edges and the finest details in the image can most possibly become the support vectors in the sparse SVM.



(a) i=1 (b) i=2 (c) i=4 (d) i=5

Fig. 1. The frequency responses of the support value operators $\boldsymbol{L}_{\mathrm{i}}$ (i=1, 2, 4, 5).



(a) γ=1000 (b) σ^2=0. 1

Fig. 2. Effect of σ^2 and γ on frequency responses of the operator $\boldsymbol{L_\alpha}$ at 7×7 center.

Figure 2 shows the effect of σ^2 (Gaussian kernel) and γ respectively on frequency responses of the support value operators $\boldsymbol{L_\alpha}$ at 7×7 neighborhood center (0, 0). From Fig. 2a, for a given γ, with the increase of the σ^2 value, the frequency response amplitude changes greatly at higher frequencies. From Fig. 2b, for a given value of σ^2, different regulation constant γ results in different frequency responses.

4 Optimal Configuration of Mapped LS-SVM

For a give image $\boldsymbol{g}$, the estimated image $\hat{\boldsymbol{g}}$ can be rewritten as follows:

$$\hat{\boldsymbol{g}} = \boldsymbol{L}_h * \boldsymbol{g} + \boldsymbol{L}_l * \boldsymbol{g} = \boldsymbol{f} * \boldsymbol{g}, \boldsymbol{f} = \boldsymbol{L}_h + \boldsymbol{L}_l \tag{11}$$

where * is the convolution, $\boldsymbol{f}$ is the operator combined of the low- and high- pass filters. The Fourier transformation of Eq. (11) is $\hat{\boldsymbol{G}} = \boldsymbol{F}\boldsymbol{G}$, where the $\boldsymbol{F}$ is the frequency response of the operator $\boldsymbol{f}$. To minimize the approximate error of the image $\boldsymbol{g}$ - $\hat{\boldsymbol{g}}$, for the mapped LS-SVM with Gaussian kernel, is to minimize the squared error,

$$\min_{\sigma,\gamma} J(\sigma,\gamma) = \overline{(\boldsymbol{F}\boldsymbol{G} - \boldsymbol{G})}(\boldsymbol{F}\boldsymbol{G} - \boldsymbol{G}) = (\boldsymbol{F} - 1)\overline{(\boldsymbol{F} - 1)}\boldsymbol{G}\overline{\boldsymbol{G}} \tag{12}$$

As the $\boldsymbol{G}$ is Fourier transformation of original image $\boldsymbol{g}$, Eq. (12) equals to minimize the $(\boldsymbol{F}-1)\overline{(\boldsymbol{F}-1)}$, which is dependent of the σ^2 and γ.

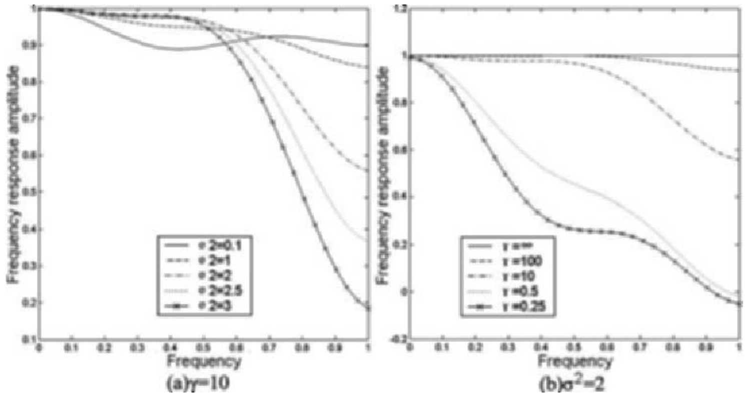


Fig. 3. Effect of the parameter σ^2 and γ on frequency responses of the operator $\boldsymbol{f}$

Figure 3a and 3b show the effect of the parameter σ^2 and γ on frequency responses of the operator $\boldsymbol{f}$, respectively, where the size of R×C is 7×7.

From Fig. 3, the regression estimation performance of the mapped LS-SVM with Gaussian kernel is significantly dependent of the configuration of parameter σ^2 and γ. When the γ is large enough ($\gamma \to \infty$), the result corresponds to the least squared error. When the γ is a specified value, there exists an optimal σ^2 (0.62) to minimize the $(\boldsymbol{F}-1)\overline{(\boldsymbol{F}-1)}$.

5 Conclusions

In this paper, determination of the kernel parameters is investigated. For the mapped LS-SVM with Gaussian kernel, the optimal σ^2 value exists, and the γ is directly determined by the frequency content of the image, which is verified by the image intensity regression estimation experiments. Mathematical analysis in the general cases is a topic in our future research.

References

Broomhead DS, Lowe D (1988) Multivariable functional interpolation and adaptive networks. Complex Systems 2: 321-355.

Martin JK, Hirschberg DS (1996) Small sample statistics for classification error rates I: Error rate measurements. (Tech. Rep. 96-21, by Department of Information and Computer Science, UC Irvine).

Smola AJ (1998) Learning with Kernels, Ph.D. thesis, Technische University at Berlin.

Steinwart I (2001) On the influence of the kernel on the consistency of support vector machines. J Machine Learning Research 2: 67-93.

Suykens JAK, Vandewalle J (1999) Least squares support vector machine classifiers. Neural Processing Letters 9: 293-300.

Tipping ME (2001) Sparse Bayesian learning and the relevance vector machine. J Machine Learning Research 1: 211-244.

Vapnik V (1995) The nature of statistical learning theory. Springer-Verlag, Berlin.

Zheng S, Tian JW, Liu J, Xiong CY (2004) Novel algorithm for image interpolation. Optical Engineering 43: 856-865.

Classifying Unlabeled Data with SVMs

Tao Wu, Hanqing Zhao

Dept. of Automatic Control, National University of Defense Technology Changsha
410073, Hunan, China
wutao.nudt@hotmail.com,
zhaohanqing@yahoo.com.cn

Abstract. SVMs have been used to classify the labeled data. While in many applications, to label data is not an easy job. In this paper, a new quadratic program model is presented so that SVMs can be used to classify the unlabeled data. Based on this model, a new semi-supervised SVM is also presented. The experiments show that the new semi-supervised SVM can be used to improve the correct rate of classifiers by introducing the unlabeled data.

Keywords: Support Vector Machine(SVM), Quadratic Programming, Semi-supervised Learning

1 Introduction

Support vector machines (SVMs), a supervised machine learning technique, have been shown to perform well in multiple areas, especially in pattern classification. Here, a supervised machine learning technique requires that all the training data should be labeled.

However, to label data is not always an easy job. It may be expensive such as some problems in bioinformatics, or it may be boring such as labeling the positive data in classifying web pages. Thus, there has been a growing interest in the use of unlabeled data for enhancing classification accuracy in supervised learning settings. This is called semi-supervised learning. Cohen et al. investigate the effect of unlabeled data on generative classifiers in semi-supervised learning. They characterize situations where unlabeled data cannot change estimates obtained with labeled data, and argue that such situations are unusual in practice (Cohen et al. 1998). It is believed that SVMs can also play an important role in semi-supervised learning just as they have done in supervised learning.

SVM is the first quadratic program model of classifying labeled data. In this paper, a quadratic program model of classifying unlabeled data will be presented. We do not call this method as clustering because this method is more suitable to divide the unlabeled data into two classes just as traditional SVMs do. Based on this model, it comes naturally to extend it to a new algorithm of semi-supervised learning.

2 Quadric Program Problem of Classifying Unlabeled Data

2.1 Partial Related Works

One of the first semi-supervised algorithms was applied to web pages classification (Blum and Mitchell 1998). From then on, many applications demonstrated that using labeled data could lead to the performance improvement over learning solely based on labeled data. Since there has been so many works in this field and an extensive review of existing techniques was presented by M.Seeger (Seeger 2001), only works that have very direct relations with our method are presented here.

Bennett and Demiriz (Bennett et al. 1998) treat datasets which are already partially labeled, formulate the semisupervised support vector machine (S3VM) as a mixed integer program (MIP). Their formulation requires the introduction of a binary variable for each unlabeled data point in the training set. This makes the problem difficult to solve for large unlabeled data. To overcome this difficulty, Fung and Mangasarian present a concave minimization model , a concave semi-supervised support vector machine(VS3VM) , which is solved by a successive linear approximation algorithm so that can handle large unlabeled datasets (with a thousand points) and solve the semi-supervised problem in a considerably shorter time. Furthermore, they present an algorithm, clustered VS3VM (CVS3VM), to cluster the data. This method includes two phases: first, use k-Median clustering to find k cluster centers and select a small subset surrounding each cluster center to be labeled by oracle or expert, then use VS3VM algorithm to classify the labeled-unlabeled dataset (Fung and Mangasarian 2001). All of these methods are in the frame of discrimination and all of them need to label some unlabeled data first and then semi-supervised problem could be solved in the supervised learning frame. Thus, these two-phase methods will take much time. Furthermore, two-phase method is not a natural method. If people want to classify some unlabeled data, they can get the hyperplane directly according to the distribution of the data. Does it possible to classify the unlabeled data in one phase just as people do? Can we have other methods to construct S3VM with lower computational complexity?

2.2 Primal Problem

Suppose $X = \{x_i\}_{i=1,\cdots,n}$ is an unlabeled data set. Previous works show that the key to solve semi-supervised problem is how to list constraint conditions for the unlabeled data. Different constraint conditions will lead to produce different models. In this paper, the constraint conditions for unlabeled data can be described as follows:

For any unlabeled data x_i, $\ni\ r_i \geq 0,\ s_i \geq 0,\ s.t.$

$$w^T \cdot x_i + b + r_i \geq 1 \quad \text{and} \quad w^T \cdot x_i + b - s_i \leq -1 \tag{1}$$

Notice that:

- If $y_i = 1$, then $w^T \cdot x_i + b \geq 1$, so $r_i = 0$, s_i can be any enough large number that (1) can be hold;
- If $y_i = -1$, then $w^T \cdot x_i + b \leq -1$, so $s_i = 0$, r_i can be any enough large number that (1) can be hold.

Now, we can get a classifier by solving the following problem as traditional SVMs as:

$$Min\ \Phi(w,b;r,s) = \frac{1}{2}(w^T \cdot w) + Cr^T \cdot s \tag{2}$$

Here, $r = (r_1,\cdots,r_n)^T$, $s = (s_1,\cdots,s_n)^T$, $r_i \geq 0, s_i \geq 0, C > 0$. Obviously, $\Phi(\cdot) \geq 0$.

As we know that the margin to be maximized in the frame of SVM is equal to $2/\|w\|$, which requires that w cannot be zero. If $b \neq 0$, we can always construct an optimal solution with $w = 0$ while the condition (1) is hold.

For example, let $b = 1$, $w = (0,\cdots,0)^T$, $r = (0,\cdots,0)^T$, $s = (2,\cdots,2)^T$, then

$$\Phi(w,b,r,s) = 0 \cdot$$

Obviously, it will be the optimal solution of (2). However, it is just a trivial solution: all the data are labeled as positive ones. Of course this is not the solution we want to obtain.

To alleviate this problem, $b = 0$ is necessary to be set. Because if $b = 0$ and $w = 0$, we will have:

$$r \geq 1 \text{ and } s \geq 1$$

Thus, $r^T \cdot s > 0$. This time, $\Phi(\cdot)$ cannot be the minimum if C is big enough.

$b = 0$ means the optimal hyperplane would pass the original point, so the training data should be centered at first and then problem (2) could be solved with the following constraint conditions:

$$w^T \cdot x_i + r_i \geq 1 \quad \text{and} \quad w^T \cdot x_i - s_i \leq -1 \tag{3}$$

This is called a primal problem of classifying unlabeled data.

2.3 Dual Problem

To solve the primal problem above, we can construct the dual problem as follows:

Let

$$L(w,b,r,s;\alpha,\beta,u,v)=\Phi(w,b;r,s)-\sum_i \alpha_i(w^T x_i + r_i - 1)-\sum_i \beta_i(-w^T x_i + s_i - 1)$$
$$-\sum_i u_i r_i - \sum_i v_i s_i \tag{4}$$

For $\frac{\partial L}{\partial w}=0$, $\frac{\partial L}{\partial b}=0$, $\frac{\partial L}{\partial r}=0$, $\frac{\partial L}{\partial s}=0$, we have:

$$w=\sum_i(\alpha_i-\beta_i)x_i \tag{5}$$

$$r_i=(\alpha_i+u_i)/C \tag{6}$$

$$s_i=(\beta_i+v_i)/C \tag{7}$$

Substitute (5) ~ (7) into (4), we can get the dual problem as follows:

$$Max\ W(\alpha,\beta,u,v)=-\frac{1}{2}\sum_{i.j}(\alpha_i-\beta_i)(\alpha_j-\beta_j)<x_i,x_j>+\sum_i\alpha_i+\sum_i\beta_i$$
$$-\frac{1}{C}\sum_i\alpha_i\beta_i-\frac{1}{C}\sum_i\alpha_i v_i-\frac{1}{C}\sum_i u_i\beta_i-\frac{1}{C}\sum_i u_i v_i \tag{8}$$

s.t.

$$\alpha_i\geq 0,\ \beta_i\geq 0, u_i\geq 0, v_i\geq 0 \tag{9}$$

Now the computation cost of this method can be evaluated: If there are n unlabeled data to be classified, there are *4n* constraint conditions while traditional SVMs only have *n* constraint conditions. So the computation cost of this method classifying n unlabeled data is no more than that of the traditional SVMs classifying *4n* labeled data.

Obviously, it can be extended to nonlinearly case by introducing kernel map. At last, the hyperplane can be obtained by solving the following quadratic programming:

$$Max\ W(\alpha,\beta,u,v)=-\frac{1}{2}\sum_{i.j}(\alpha_i-\beta_i)(\alpha_j-\beta_j)k(x_i,x_j)+\sum_i\alpha_i+\sum_i\beta_i$$
$$-\frac{1}{C}\sum_i\alpha_i\beta_i-\frac{1}{C}\sum_i\alpha_i v_i-\frac{1}{C}\sum_i u_i\beta_i-\frac{1}{C}\sum_i u_i v_i \tag{10}$$

with constraint condition (9).

Notice that there is a difference between kernel in (10) and that in traditional SVMs since we have mentioned that the data should be centered before classifying the unlabeled data. This means the kernel should be an inner product of two centered vectors. The new kernel can be computed as follows:

Let $k(x_i,x_j)$ is the inner product between $\phi(x_i)$ and $\phi(x_j)$, the center of data can be expressed as:

$$\overline{\phi}(x) = \frac{1}{n}\sum_{i=1}^{n}\phi(x_i)$$

The corresponding kernel of centered data is:

$$\widetilde{k}(x_i, x_j) =< \phi(x_i) - \overline{\phi}(x), \phi(x_j) - \overline{\phi}(x) >$$

$$= k(x_i, x_j) - \frac{1}{n}\sum_{j=1}^{n}k(x_i, x_j) - \frac{1}{n}\sum_{i=1}^{n}k(x_i, x_j) - \frac{1}{n^2}\sum_{1\le i,j\le n}k(x_i, x_j) \tag{11}$$

The classifier can be represented as:

$$f(x) = \sum_{i=1}^{n}(\alpha_i - \beta_i)\widetilde{k}(x, x_i) \tag{12}$$

Figure 1 is an artificial example. Here, the data are from Matlab SVM ToolBox presented by Steve Gunn. The results show that this method is satisfactory. In the left figure, the kernel is selected as “linear”, which means to classify the unlabeled data in the input space. The data are divided into two parts according to their distances to the hyperplane. This result is not easy to be obtained by traditional clustering algorithms because most traditional clustering algorithms are based on the distances between the data. So we believe that this method will be a useful supplement to traditional clustering algorithms.

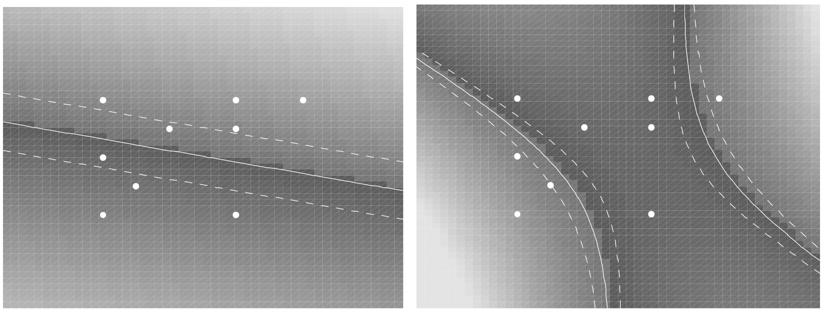


Fig. 1. SVMs for classifying the unlabeled data: Linear kernel (left) and RBF kernel(right, $\sigma = 1$)

3 Semi-supervised Learning

As we have mentioned that many previous works are trying to label a part of data and then use semi-supervised learning methods to cluster data. On the contrary, we present a clustering algorithm in advance and now, it naturally comes to use this algorithm for semi-supervised learning.

In fact, the only difference between the clustering algorithm and the semi-supervised learning algorithm in the frame of SVM is that there are some extra constraint conditions for labeled data:

For any labeled data, the constraint condition is just as that in traditional SVMs:

$$y_i(wx_i + b) \geq 1.$$

To combine it with clustering algorithm, we need center the data too. So the constraint conditions are turned to be: $y_i(wx_i) \geq 1$. We also turn to solve the dual problem to obtain the classifier. The obtained classifier can be represented as follows:

$$f(x) = \sum_{i \in LX} \alpha_i y_i \tilde{k}(x, x_i) + \sum_{i \in UX} (\beta_i - \gamma_i) \tilde{k}(x, x_i) \tag{13}$$

Here, *LX* is denoted as labeled data, and *UX* is denoted as unlabeled data. $(\alpha_i, \beta_i, \gamma_i)$ is the optimal solution of the following problem:

$$\begin{aligned} Max\, L(\alpha,\beta,\gamma,\mu,\nu) = &-\frac{1}{2}\sum_{i,j \in UX}(\beta_i - \gamma_i)(\beta_j - \gamma_j)\tilde{k}(x_i, x_j) - \frac{1}{2}\sum_{i,j \in LX}\alpha_i\alpha_j y_i y_j \tilde{k}(x_i, x_j) \\ &- \sum_{i \in LX}\sum_{j \in UX}\alpha_i(\beta_j - \gamma_j) y_i \tilde{k}(x_i, x_j) + \sum_{i \in LX}\alpha_i \\ &+ \sum_{i \in UX}(\beta_i + \gamma_i) - \frac{1}{C}\sum_{i \in UX}(\beta_i\gamma_i + \beta_i\nu_i + \gamma_i\mu_i + \mu_i\nu_i) \end{aligned} \tag{14}$$

s.t.

$$\alpha_i, \beta_i, \gamma_i, \mu_i, \nu_i \geq 0$$

$\tilde{k}(\cdot,\cdot)$ is defined as (11) defined.

To test the performance of our method, we choose two benchmarks from semi-supervised challenge on NIPS'2001. One is "neural image" and another is "horse shoe". "Neural image" have 530 training data and 2710 test data, while "horse shoe" has 400 training data and 800 test data.

In the experiments, we choose the last 50 training data as extra unlabeled data and different number of training data as labeled: 2, 10, 20, 30, 40 and 50. Here, 2 means every class has only one label data and 10 means there are 10 labeled data (first 10 training data); 20 means first 20 labeled data are training data, and so on. C-SVM means classical SVMs and S-SVM means semi-supervised SVM presented in this paper. RBF kernel is chosen for all the SVMs.

Usually, unlabeled data is useful to improve the correct rate of the classifier when there are only a few labeled data. However, some contrary cases happened (See column 2 in table 2). Maybe it is caused that the horse shoe data have higher dimensions: neural image data have 12 features while horse shoe data have 25 features. In high dimension spaces, the data may have so long distances that the distances between the data in the same class could be longer than that in different classes.

To overcome this drawback, some new distances, such as the geodesic distances, can be used as a substitute for Euclidean distance just as mentioned in (M.Belkin and P.Niyogi, 2004). The new distances can be combined with kernel’s construction, but this will be our further research.

There is another problem that should be mentioned: the classifier is very sensitive to the value of C. How to select a suitable value of C is also an important problem that need to be studied.

Table 1. 50 unlabeled data and different number of labeled data for neural image data (σ=3,C=200)

#Labeled	2	10	20	30	40	50
C-SVM	946	1581	1935	2048	1813	1788
S-SVM	1540	1685	1831	1806	1850	1837

Table 2. 50 unlabeled data and different number of labeled data for horse shoe data (σ=10,C=200)

#Labeled	2	10	20	30	40	50
C-SVM	485	499	478	437	476	485
S-SVM	487	491	496	490	508	507

4 Conclusions

In this paper, we present a new quadratic program model to classify the unlabeled data. May be this is the first model to cluster the unlabeled data by solving a quadratic programming in the frame of SVMs. The method enjoys its low computation cost: the computation cost of our method to classify n unlabeled data is almost equal to that of classical SVMs to classify $4n$ labeled data. Based on this method, we present another semi-supervised learning method. Experiments show that this method can improve the correct rate of the classifier when the labeled data is very few.

Acknowledgments

This work is supported by National Nature Science Foundation (No.60234030).

Reference

Belkin, M., Niyogi, P.(2004), “Semi-Supervised Learning on Riemannian Manifolds”, *Machine Learning 56*, pp.209-239.

Bennett, K. P. and Demiriz, A.(1998), Semi-supervised support vector machines”, D. A. Cohn M. S. Kearns, S. A. Solla, editor, *Advances in Neural Information Processing Systems* -10-, pp.368-374, MIT Press, Cambridge, MA.

Blum, A. and Mitchell, T.(1998), “ Combining labeled and unlabeled data with co-training”, *In COLT: Proceeding of the workshop on computational learning Theory*. Morgan Kaufmann Publishers.

Cohen, I., Cozman, F.G., and Bronstein, A.(2002), “The effect of unlabeled data on generative classifiers, with application to model selection” , *Technical Report, HP Laboratories Palo Alto HPL-2002-140*

Fung, G. and Mangasarian, O. L. (2001), “Semi-Supervised Support Vector Machines for Unlabeled Data Classification”, *Optimization Methods and Software ,* vol. 15, pp. 29–44.

Nigam, K., McCallum, A., Thrun, S., Mitchell, T.(2000), “Text Classification from Labeled and Unlabeled Documents Using EM”. *Machine Learning.* pp. 103–134

Seeger, M.(2001), Learning with Labeled and Unlabeled Data”, *Technical Report. Edinburgh University, UK.* Available from Http://citeseer.ist.psu.edu/seeger01learning. html

Part XVII

Robotics

Car Auxiliary Control System Using Type-II Fuzzy Logic and Neural Networks

Peipei Fang and Yan-Qing Zhang*

Department of Computer Science
Georgia State University
Atlanta, GA 30302-3994 USA
*Contact author: yzhang@cs.gsu.edu

Abstract. In this paper, a Type-II fuzzy and neural network system is introduced to car auxiliary control system for inexperienced drivers. The approach applies Type-II fuzzy logic to generate the training data set for the neural network. The Type-II fuzzy logic is launched by the component called the fuzzy controller. With Type-II fuzzy logic and neural network system, the auxiliary system can manipulate more driving conditions so that enables the neural network to predict more reliably. The results of the approach is compared with Type-I fuzzy/neural system and discussed in detail.

Keywords: Type-I fuzzy logic, Type-II fuzzy logic, neural networks, auxiliary control system

1 Introduction

Lack of driving experience can easily cause serious traffic accidents if there are no instructions from any experienced or professional drivers. Several applications have been considered and designed to handle path searching, obstacle detection and avoidance. An auxiliary control system is desired to help new drivers drive without risk and learn to make proper decisions. Fuzzy logic is the most used approach to offer the controlling support in this area (Sousa and Kaymak 2002) and (Altrok et al 1992). Some of the applications apply neural networks (Nguyen and Widrow 1989) to increase the accuracy of the result successfully, but limitedly.

We designed an auxiliary control system using Type-II fuzzy logic and neural network. This system accepts the data sets from some peripheral devices and determines what necessary actions should be taken. The determinations will be presented to drivers. Some sensors are assumed to be installed on the body of the car pointing to different directions so that we can obtain the necessary information of road ahead and aside. With the delivered information from the sensors and the speed of the car, this auxiliary control system can determine to give the driver a warning or not based on the result of the fuzzy and neural system.

In Figure 1, the driving conditions are obtained by the sensors and preprocessed in the data preprocesser. The main functionality of the preprocesser is to create the crisp inputs for the auxiliary control system. With the required inputs, the fuzzy controller applies the fuzzy logic to the inputs. We employ the neural network to generate the outputs. These outputs are then transmitted into other devices so that necessary or critical actions are displayed by sending out the warning.

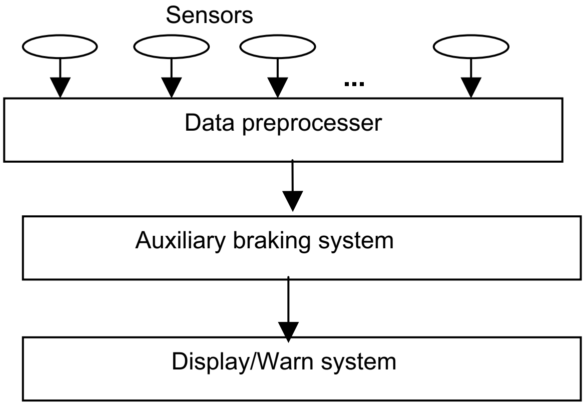


Figure 1. Auxiliary control system.

Section 2 focuses on Type-I and Type-II fuzzy logic and the fuzzy controller. Section 3 discusses the neural network which is the learning part of the auxiliary system. Simulation and comparison are presented in Section 4. Finally the conclusion is introduced.

2 Fuzzy Controller

The fuzzy controller is a tutor to the neural network. It is developed to provide the training data set by referring to the rule base containing fuzzy if-then rules. In our first approach, it contains linguistic variables and defuzzificating functionality (Jang et al 1996) and (Zimmermann 1991). In the fuzzy controller, the input and output data are treated as linguistic variables for the raw data to be processed to gain the related fuzzy sets.

2.1 Define fuzzy sets

For a car under driving condition, there are three basic parameters need to be considered, such as *speed*, *distance*, and *direction*. Speed is the current velocity of the car; distance is between the car and the obstacle in front, such as vehicles, humans, or other objects; direction is the position of the obstacle in front of the car. With these three parameters, we can therefore depict the road condition as well. First of all, the speed of car is divided into five intervals within the range of [0, 100], described in Table 1.

Table 1. Fuzzy Sets for Speed.

SP	**Speed**	**Interval**
HA	halt	0 – 25
SL	slow	0 – 50
MID	middle	25 – 75
FS	fast	50 – 100
TOS	topspeed	75 – 100

Table 2. Fuzzy Sets for Distance

DIS	Distance	Interval
AC	accidental	0 – 5
CR	critical	0 – 10
CL	close	5 – 20
FA	far	10 – 30
INF	infinite	20 – 30

Suppose all of the sensors are separated into three groups. Each group includes five sensors pointing to one direction out of three different directions. Three directions are: *Left*, *Middle*, and *Right*. Based on the direction set, the sensors pointing to a particular direction return integer values between 0 and 10 to specify the position of the obstacles on the road need to be considered or avoided. The distance of obstacle is divided into five intervals within the range of [0, 30], described in Table 2. The outputs of the fuzzy controller are two linguistic variables, *change of speed* and *change of direction*. Let us determine the range of the parameter of *change of speed*. It is within the range of [0, 40] and divided into five intervals. The range of the parameter of change of direction is within the range of [0, 40] and divided into five intervals.

Table 3. Fuzzy Sets for Outputs.

COS	Change of speed	Interval
STB	Strong brake	0 – 10
SOB	Soft brake	0 – 20
CON	Continue	10 – 30
ACC	Accelerate	20 – 40
FSP	Full speed	30 – 40
COD	**Change of direction**	**Interval**
LB	Left big	0 – 10
LS	Left small	0 – 20
ZE	Zero	10 – 30
RS	Right small	20 – 40
RB	Right big	30 – 40

2.2 Processes

The first step of the fuzzy controller is to calculate each rule in the rule base the degree to which the rule in discussion is satisfied. Before doing that, the rule base must be defined. For our auxiliary control system, there are two rule bases required because of the output contains two different aspects: *change of speed* and *change of direction*. Generally, the fuzzy controller tries to match the membership functions of the inputs with the linguistic values present in the rule base. In order to implement this, a fuzzy-and operator is applied to the and-terms of the rule, in other words, a standard minimum operator is applied.

The matching values are carried through to calculate the inference results. According to the basic knowledge of the fuzzy logic, the approach decides the minimum between the matching value and the result of the rule. And then, the fuzzy sets derived from the inference approach are used to generate the final results. A fuzzy-or operator, or maximum operator, is applied to all fuzzy sets to obtain the final result which can be transformed into a crisp value. This procedure is called defuzzification using calculating center of the area.

Since the program has two responses to a situation, which are *change of speed* and *change of direction*, the fuzzy controller is divided into two similar parts but work on two different rule bases. Once we setup the rules necessary to monitor and guide the behaviors of the car, the auxiliary control system can give appropriate responses to the driver according to the road situation. But there are still many extreme situations in which our rules are not defined sufficiently, such as U-turn and so on. Solving those problems is a time-consuming and tedious process. Fortunately, some approaches have been worked out to solve those problems (Altrok et al 1992) and (Fodor 1991).

2.3 Type–II Fuzzy Logic

The Type-II fuzzy systems still use if-then rules but participate with different antecedent and consequent sets from those of Type-I fuzzy logic systems. Figure 2 (Mendel 2001) shows us the concept on how Type-II Fuzzy logic system functions.

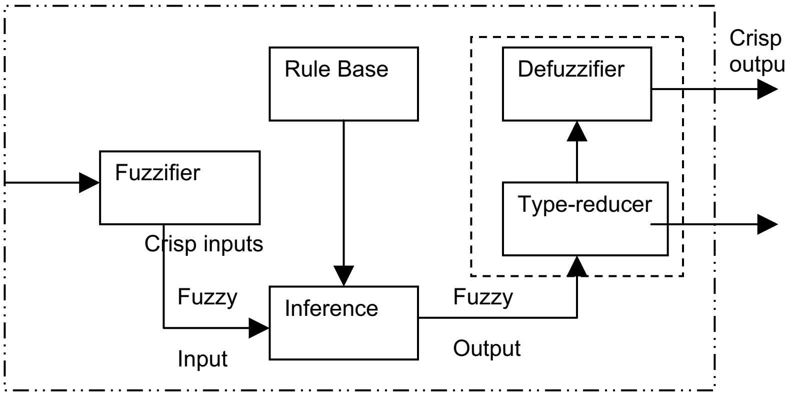


Figure 2 Type-II Fuzzy logic system

Their antecedent and consequent sets are, of course Type-II. Briefly speaking, a Type-II fuzzy set can present and deal with uncertain information much better than a Type-I fuzzy set. Type-II fuzzy sets can model and minimize the effects of the uncertainties in our fuzzy logic system (Mendel and John 2002) and (Mendel 2001). Type-I fuzzy sets can not handle uncertainties because their membership functions are crisp, which are of two-dimensional, however Type-II membership functions are of three-dimensional. We give out more details on explaining Type-II fuzzy logic. Let A is a Type-I fuzzy set, $x \in X$, then the membership grade u of x in A is $\mu_A(x)$, a crisp number of [0, 1]; Let A' is a Type-II fuzzy set, $x \in X$, then the membership grade of x in A' is $\mu_{A'}(x)$, a Type-I fuzzy set of [0, 1]. A' can be described as below:

$$A' = \{((x,u), \mu_{A'}(x,u)) \mid \forall x \in X, \forall u \in U \subseteq [0,1]\}, 0 \le \mu_{A'}(x,u)$$

$\mu_{A'}$ is the Type-II membership function, u is the membership grade of the Type-I fuzzy set. Generally, A' is in such a formula: $A' = \sum_{x \in X} \left[\sum_{u \in U} f_x(u)/u \right] / x$. The Type-II representation theorem defines that a Type-II fuzzy set can be represented as the union of its Type-II set, $A' = \sum_{j=1}^{n} A^j$. The Type-II fuzzy set output needs to be reduced to a Type-I fuzzy set by the type reducer to produce the crisp output by the defuzzifier. Details about Type-II fuzzy logic can be found in (Mendel 2001) and (Liang and Mendel 2000).

In reality, there are many other features need to be considered, such as weather, traffic-load and daylight. All of these features are time varying and mutable and are unpredictable. For our problem, we consider two important features for our auxiliary control system: weather and traffic. As mentioned before, we need to define primary membership functions to generate the membership grades in [0, 1]. Consequently, we need to define the secondary membership functions for Type-II fuzzy sets which has primary grades as the inputs. Therefore, the useful part is the footprint of uncertainty of the Type-II fuzzy set (J. Mendel and R. John 2002), (Mendel 2001) and (Liang and Mendel 2000). The secondary membership functions of Type-II fuzzy logic can be properly created by considering the primary memberships and secondary grades. If we use Gaussian functions to express the relations of them, the functions are:

$$A' = \exp(-\frac{(x-w)^2}{t}), \ w \in [w_1, w_2] \text{ and } t \in [t_1, t_2] \qquad \text{(secondary MF for Speed)}$$

$$A^{'} = \exp(-\frac{(x-w)^2}{t}) \qquad \text{(secondary MF for Direction)}$$

In the formulas, we have two rule bases to present the speed and the direction of the car: *s* and *d*. Two outputs are *change of speed* and *change of direction* presented in linguistic variables. *w* represents the uncertainty of weather and *t* represents the uncertainty of traffic-load.

3 Neural Networks

The backpropagation (Rumelhart et al 1996) and (Jang et al 1997) algorithm is facilitated to train the neural system. In our case, sigmoid function is used to squash the input data of each node into the range between 0 and 1. The fuzzy controller processes by applying the fuzzy rules, inference engine and Type-I or Type-II fuzzy logic. The neural network will be trained with the data set created by the fuzzy controller that uses Type-II fuzzy for times to reduce the error below a predefined threshold. The trained neural network is then used to support the driving system (See Figure 3).

Based on the data sets created by Type-II fuzzy system in the fuzzy controller, the neural network can offer appropriate decisions to drivers. But, the improvement of our auxiliary control system is derived from the participation of Type-II fuzzy system because of its consideration on the uncertainties of driving situations. In the training process, every input is presented to the neural network for a certain times until the neural network reduces the error to below a predefined threshold.

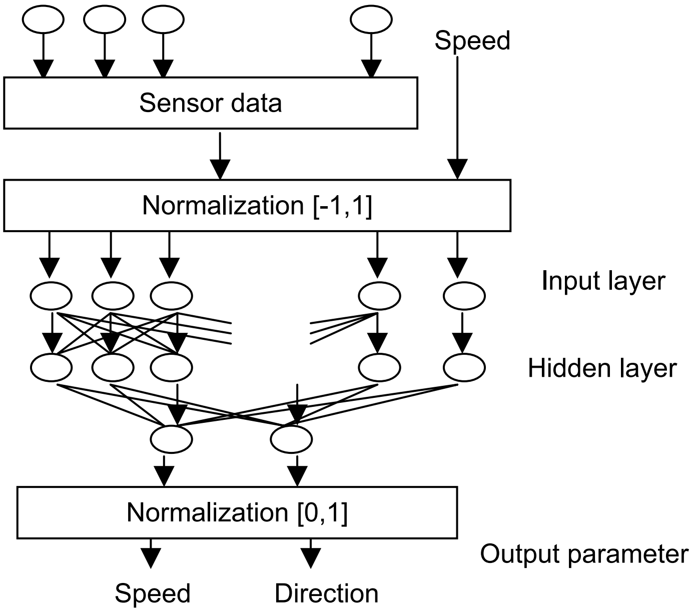


Figure 3. Neural network.

4 Simulations and Results

In our first approach, we apply the Type-I fuzzy logic which is basic to our system. As described in Section 3, we understand that it is the foundation of Type-II fuzzy logic system. After we design the Type-I fuzzy logic system, the design of Type-II fuzzy logic system can be extended from it. First of all, we need to carefully define the fuzzy sets and membership functions for speed, distance and direction.

4.1 Type-I fuzzy logic system

For the auxiliary system, there are two different components in the fuzzy controller to calculate the change of speed and change of direction respectively. Basically, we need to define two fuzzy rule bases to match the requirements to creating the outputs. The more if-then rules are in the fuzzy rule base the more accurate the results can be. According to the principle of incompatibility, even we add more than hundreds if-then rules to the base, we are not going to make things better because there are too many unpredictable features that can effect the final results. In fact, most of the rules are strongly dependent on human judgment and perception, which will reduce the precision as well (Mansour and Fox 1991). For this reason, we add those if-then rules that can describe the real situation properly and leave the rest to the neural network. We collect all these descriptions for the real situation and define the fuzzy rule bases. The following procedure is to define the fuzzy inference engine. Herein, we apply the Mamdani-FIE to map fuzzy inputs to fuzzy conclusions. Practically, we will use center of area method to simplize the defuzzification process and then get the crisp outputs that will be stored as part of the training data set.

4.2 Type-II fuzzy logic system

Type-II fuzzy sets are three-dimensional and the membership grades are fuzzy sets in the range of [0, 1]. Type-II fuzzy logic can handle the uncertainties by using the Type-II membership functions which effectively describe the uncertain linguistic values.

Consider the possibilities for Type-II fuzzy logic system, we find that (Mendel and John 2002) when the words used in the antecedents or consequents of rules have different meaning to different people; the consequents may vary according to uncertain knowledge; noisy is invoked during the Type-I fuzzy logic process; also the data to tune the Type-I fuzzy logic system parameter could be noisy. In our case, the weather and the traffic situation are time-varying features that indeed affect the decision making on driving. Though the weather degree or traffic rate can eventually be represented by measurable domains, these two features still mean different to different people. This phenomenon can easily raise the uncertainty of words. In reality, driving is not as simple as we described before. Speed, distance, and obstacle position (direction) are the most important features with which one should consider. Many external features will take parts on this stage. Therefore, we have the necessity to include the uncertainties into our auxiliary control system and make it suited for the novices. Let us concern the membership functions for the speed of the car.

We need to do some modifications so that the uncertain features discussed above can be included. The range of θ is [interval-7, interval+7]. With this parameter, the membership grade of the speed can be located within a data set. Based on this concept, we can derive the formulas for the distance between the obstacle and the car. The range of σ is [interval-3, interval+3]. The footprint then is the most important area of the membership functions. (See Figure 4).

The fuzzy rule bases and fuzzy sets of Type-II fuzzy logic are same as those of its Type-I fuzzy logic. The additional procedure is the adaptation of fuzzy type reduction. For our program, we use the centroid type reduction method to combine the Type-I fuzz outputs.

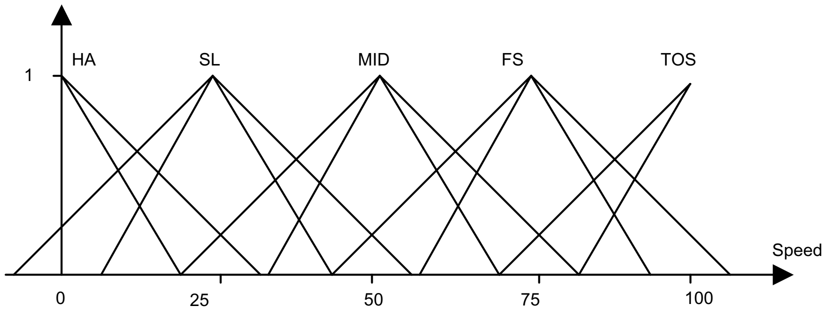


Figure 4. Type-II membership function. The region between two solid lines is so called footprint which is the most important part of membership function.

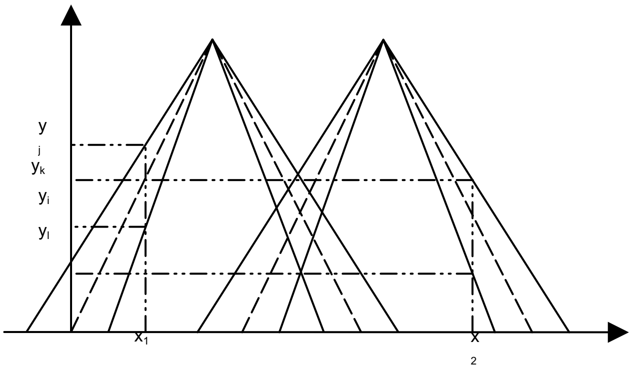


Figure 5. Type-II fuzzy logic fire strength

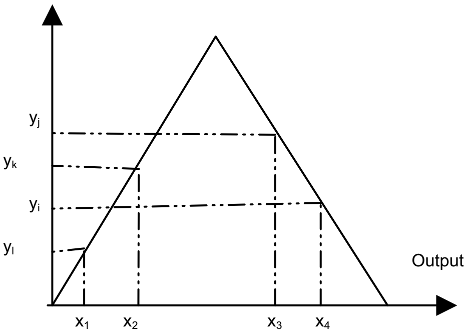


Figure 6. Type-II fuzzy logic fire strength

With Type-II fuzzy fire strength (Figure 5), we perform the type reduction. In this procedure, a *t-cnorm* is used and then the program finds the center of the fuzzy set (Karnik et al 1999). Finally we apply the defuzzification method which converts the fuzzy outputs into crisp output that is understandable (Figure 6).

$$\frac{\sum y_i \theta_i}{\sum \theta_i} \tag{1}$$

In the formula (1), $y \in Y, \theta \in \tilde{\mu}(y)$. We finally come up with another crisp output that represents the action should be taken to the car. Theoretically, a car needs to be driven for adequate times (the more the better) to obtain the raw data from the sensors, the speed meter and the actual actions taken by the experienced driver. The raw data will go through the fuzzy controller to generate the training data set by fuzzy inference engine and fuzzy rule bases. The neural network will consequently be trained with these training data to adjust the interior weights so that the network can predict the appropriate actions which should be made for newly occurred events. In our auxiliary control system, the fuzzy controller and neural network are separate.

Once the fuzzy controller is finished with generating the training data, the control will be transferred to the neural network which gets trained with the training data set and provides drivers the appropriate responses.

4.3 Results

We use randomization method to generate the raw input data such as car speed, distance between the car and the obstacle, and position of obstacle because currently we still can not apply our auxiliary control system to a real environment. The simulation program generates almost one hundred random values of speed, distance and position. We launched the first approach using Type-I fuzzy logic to create the output values and derive the decisions. Then we launched the second approach with Type-II fuzzy logic and generated the training data which was delivered to the neural network later. In the neural network, the network read the training data and applied the backpropagation algorithm to learn the data according to the parameter given at the command line. When all these processes were done, we retrieved the raw data that had been used in the first approach and launched the second approach to produce the output.

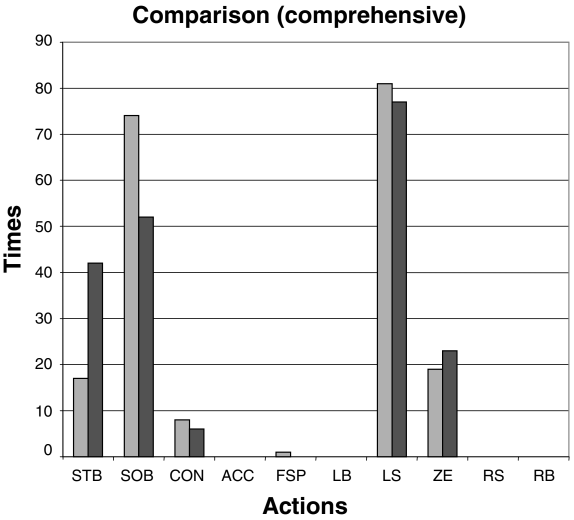


Figure 7. Comprehensive Comparison

In Figure 8, it seems that Type-II fuzzy and neural network simulator overreacts for very low driving speed comparing with another system. In Figure 9, Type-II fuzzy and neural network simulator and Type-I fuzzy simulator perform approximately. But we still can tell that the former acts more carefully and humbly in even safe situations. In Figure 10, the difference is obvious. The main difference bases on the cases where speed is high. At this time, the Type-II fuzzy and neural network simulator comes up with more reasonable and reliable result: *STB (strong brake)*. We discover this importance in our last comparison (Figure 11) where the conditions are high speed and short distance between the car and the obstacle. Now we find that the Type-II fuzzy and neural network simulator definitely outrun the Type-I fuzzy simulator. It takes much more *STB (strong brake)* actions, which is necessary and safe because the driver is driving fast and the obstacle distance is shrinking quickly. The result illustrates that the Type-II fuzzy and neural network simulator works more like human experts.

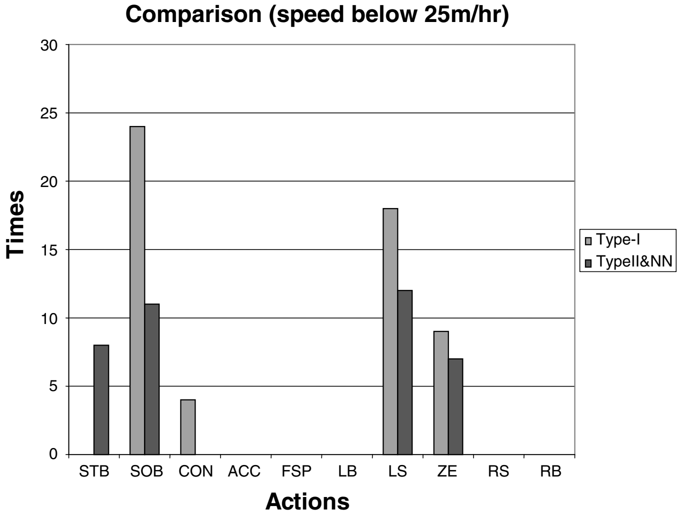


Figure 8. Comparison for low speed

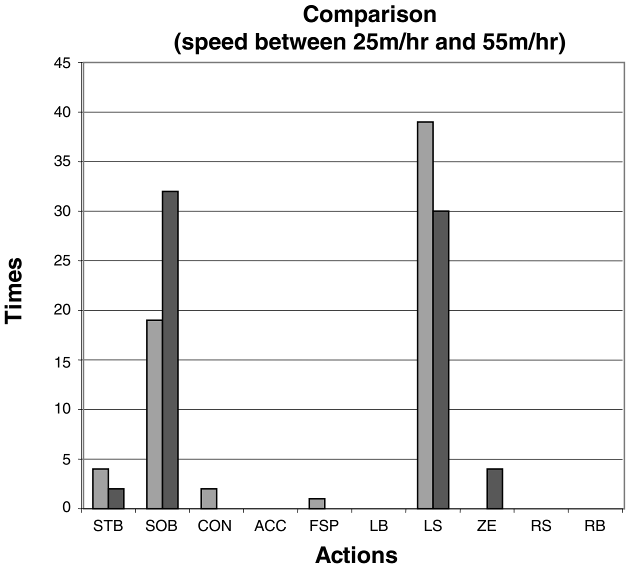


Figure 9. Comparison for middle speed

Type-II fuzzy and neural network simulator have more *STB (strong brake), SOB* (soft brake) actions and approximately same *LS* (small left) actions than Type-I fuzzy simulator does, according to most of our comparisons at: low speed, middle speed, and high speed. Especially, the Type-II system seems to be able to provide the driver more reliable and more reasonable warnings in the severe situations than Type-I fuzzy simulator does. More *SOB (soft brake)* actions are taken by the Type-II fuzzy and neural network system in regular cases. More *STB (strong brake)* actions are also taken by it when the speed is high. The Type-II fuzzy and neural network system works much better than the Type-I fuzzy system in the serious and bad traffic situations. In the case of low speed, the Type-I fuzzy system can work much better in stead. Nevertheless, in this case a new driver also can handle most of the events successfully. So, when the speed is very slow, the driver can definitely count on himself/herself.

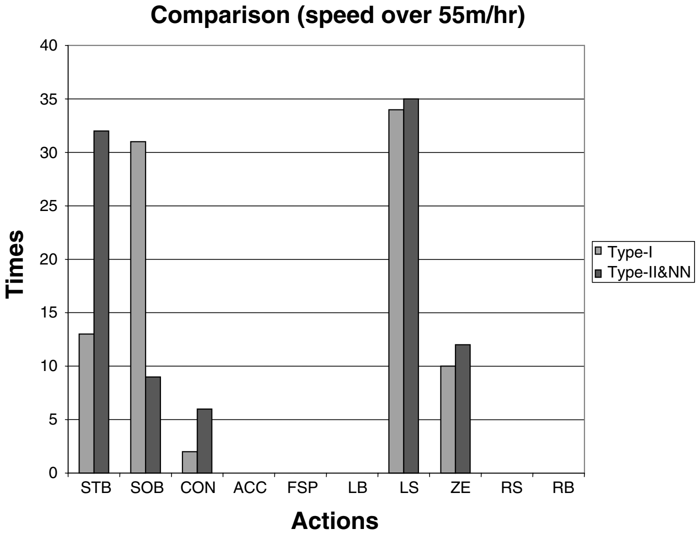


Figure 10. Comparison for high speed

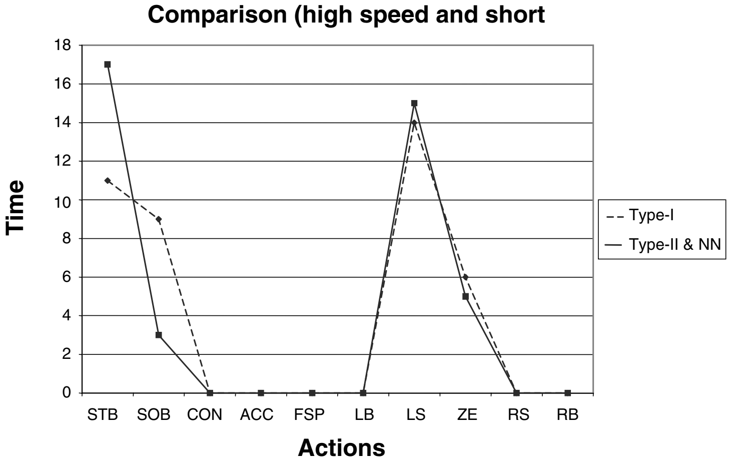


Figure 11. Result comparison for high speed with short distance

5 Conclusions

From the graphs and those described above, we can draw the conclusion that the auxiliary control system using Type-II fuzzy and neural network can provide more reasonable and reliable responses in the case that the speed is fast and the obstacle distance is short than the Type-I fuzzy approach does, which is exactly what we desire for new and inexperienced drivers. Meanwhile, the Type-II fuzzy and neural network approach also recommends more modest and courteous behaviors than the Type-I fuzzy approach does, which is undoubtedly good for drivers.

For these two reasons, the Type-II fuzzy and neural network approach is more capable of helping our new and inexperienced drivers out of the probable accidents than the Type-I fuzzy logic approach. The former system can come up with more reliable warnings for drivers' considerations, especially when they are stuck in bad driving conditions where the visibility decreases and the risk possibility increases. Thought each approach exceeds another one in different cases, we can still combine them together into our auxiliary control system. When the speed is low, the Type-I fuzzy approach will control and provide the warnings. Otherwise, the Type-II fuzzy and neural network approach will take over the control and "watch out" for drivers.

6 References

Altrok, C. von, Krause, B. and Zimmerman, H. J., "Advanced fuzzy logic control of a model car in extreme situations," *Fuzzy Sets and Systems*, 48(1):41-52, 1992.

Fodor, J.C., "On fuzzy implication operators," *Fuzzy Sets and Systems*, 42:293-300, 1991

Jang, J.-S. R., Sun, C.-T. and Mizutani, E., *Neuro-Fuzzy and Soft Computing*, Prentice Hall, NJ, 1997.

Karnik, Nilesh N., Mendel, Jerry M. and Liang, Qilian, "Type-2 Fuzzy Logic Systems," *IEEE Trans. On Fuzzy System*, Vol. 7, No.6, pp. 643-658, 1999.

Liang, Q. and Mendel, J., "Interval Type-2 Fuzzy Logic Systems: Theory and Design," *IEEE Trans. On Fuzzy System*, Vol. 8, No.5, pp. 535-550, 2000.

Mansour, N. and Fox, G. C., "A hybrid genetic algorithm for task allocation in multicomputers", *Proceeding of Fourth International Conference on Genetic Algorithms*, pp. 466-473, July, 1991

Mendel, J. and John, R., "Type-2 Fuzzy Sets Made Simple," *IEEE Trans. On Fuzzy Systems*, Vol. 10, No.2, pp. 117-127, 2002

Mendel, J. M., "On the Importance of Interval Sets in Type-2 Fuzzy Logic Systems," *Proceedings of Joint 9th IFSA World Congress and 20th NAFIPS Int'l. Conf.*, Vancouver, British Columbia, Canada, July 25-28, 2001.

Mendel, J., *Uncertain Rule-Based Fuzzy Logic System: Introduction and New Directions*, Prentice-Hall: NJ, 2001.

Nguyen , D. and Widrow, B., "The truck backer-upper: An example of self-learning in neural networks," Proc. Of the Int. Joint Conference on Neural Networks, vol.2, pp. 357-264, 1989.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J., "Learning internal representations by error propagation," In: D.E. Rumelhart and J.L. McClelland (Eds.), *Parallel Distributed Processing*, MIT Press, Cambridge, 1996.

Sousa, Joao M. C. and Kaymak, Uzay, "Fuzzy logic control," *Fuzzy Decision Making in Modeling and Control*, Singapore, World Scientific Publishing Co. Pte. Ltd., 2002

Zimmermann, H.-J., *Fuzzy Set Theory – and its Applications*, Kluwer Academic, Boston, 1991.

Evolving Neural Controllers for Collective Robotic Inspection

Yizhen Zhang[1], Erik K. Antonsson[1] and Alcherio Martinoli[2]

[1] Engineering Design Research Laboratory, California Institute of Technology,
Pasadena, California 91125, USA
yizhen@caltech.edu erik@design.caltech.edu

[2] Swarm-Intelligent Systems Research Group, École Polytechnique Fédérale de
Lausanne (EPFL), CH-1015 Lausanne, Switzerland
alcherio.martinoli@epfl.ch

Summary. In this paper, an automatic synthesis methodology based on evolutionary computation is applied to evolve neural controllers for a homogeneous team of miniature autonomous mobile robots. Both feed-forward and recurrent neural networks can be evolved with fixed or variable network topologies. The efficacy of the evolutionary methodology is demonstrated in the framework of a realistic case study on collective robotic inspection of regular structures, where the robots are only equipped with limited local on-board sensing and actuating capabilities. The neural controller solutions generated during evolutions are evaluated in a sensor-based embodied simulation environment with realistic noise. It is shown that the evolutionary algorithms are able to successfully synthesize a variety of novel neural controllers that could achieve performances comparable to a carefully hand-tuned, rule-based controller in terms of both average performance and robustness to noise.

1 Introduction

Autonomous mobile robots share important characteristics with simple biological systems: robustness, simplicity, small size, flexibility and modularity. Each individual is rather simple with limited local sensing and actuating capabilities, while as a group they can accomplish difficult global tasks in dynamic environments, without any external guidance or centralized control [3].

Design and control of such a robot swarm are difficult mainly because their group behavior is an emergent property of their mutual interaction and that with the environment, which become a distributed dynamical system due to independent parallel actions of different individuals. Since the robots only have partial perceptions based on crude noisy sensors, limited computational capabilities and energy budget, managing the robots to solve a global task under such constraints presents significant technical challenges. This is especially true because human intelligence is specialized in individuals and centralized control, instead of the collective intelligence shown in nature.

Evolutionary robotics [13] is a new and promising technique for automatic design of such autonomous robots. Inspired by nature, evolutionary robotics makes use of tools such as neural networks and evolutionary algorithms.

Inspired by biological neural networks, Artificial Neural Networks (ANN) have been a powerful computational tool widely applied in science and engineering [8]. They are often used to implement robot controllers because of their light computational requirements and nonlinear basic elements, properties that allow for real-time control and, potentially, modular implementation of complex perception-to-action functions. ANN can be designed and trained using various methods, including those based on evolutionary computation [14, 18]. As opposed to optimization of behavior-based controllers, the key feature of ANN evolution is that, the genotypical searching space is less constrained by ANN models and the resulting phenotypical solution directly shapes the robot behavior as a whole.

Evolutionary algorithms [2, 6, 12], loosely inspired from biological evolutionary processes, have gained considerable popularity as tools for searching vast, complex, deceptive, and multi-modal search spaces with little domain-specific knowledge. In recent years, they have found natural applications in the automatic synthesis of artificial neural networks for intelligent agents [15]. Evolutionary algorithms allow co-evolution of the network architectures as well as the weights within task-specific design constraints. Multiple design objectives can be expressed as fuzzy preferences and aggregated into the fitness function with different weights and trade-off strategies, which can be tuned to evolve solutions with different engineering design trade-offs [1, 16]. As stochastic optimization methods, evolutionary algorithms are also good at working in noisy environments and searching for robust solutions.

Evolution of both feed-forward and recurrent neural controllers are considered in this paper. Only synaptic weights are evolved if the ANN topology is pre-defined; otherwise the network structure and synaptic weights are simultaneously evolved, where the candidate solutions can be modified by adding or deleting hidden neurons, establishing or removing connections between any two neurons, as well as changing values of the synaptic weights.

The efficacy of the evolutionary methodology is demonstrated in the framework of a realistic case study concerned with collective robotic inspection of regular structures. The controllers are optimized in a realistic, sensor-based, embodied simulator, then downloaded to real hardware. If the embodied simulator is faithful enough for the target hardware platform, evolved controllers can be easily transfered to real robots [11]. Homogeneity of the robot team is enforced here to limit the search space, achieve scalability, and bypass the credit assignment problem typically arising in distributed systems consisting of individuals using only local information [7, 17].

The performance of the evolutionary results is compared with that of a hand-coded, rule-based controller carefully tuned for the same task. It will be shown that the evolutionary algorithm appears powerful and promising for automatic synthesis of novel ANN controllers, requiring little prior domain knowledge or ANN structural information. The evolved solutions can serve as good starting points for further study and optimization by human engineers.

2 Evolutionary Methodology

Based on the well-known principles of genetic algorithms (GA) and evolutionary strategies (ES), our evolutionary optimization loop uses real numbers to encode synaptic weights, variable chromosome length to evolve ANN structures, traditional roulette wheel selection with fitness scaling, and both crossover and mutation genetic operations to improve candidate solutions [1].

2.1 Encoding of Artificial Neural Networks

The ANN synaptic weights are directly encoded as a sequential vector of real numbers, as shown in Fig. 1. The vector (chromosome) length is static if the network structure is *a priori* determined, where a fully connected ANN is usually assumed and the fixed chromosome length can be computed as follows:

$$n_c = \begin{cases} (1+n_i)n_o & \text{if } n_h = 0 \ \& \text{ feed-forward} \\ (1+n_i)n_h + (1+n_h)n_o & \text{if } n_h > 0 \ \& \text{ feed-forward} \\ (1+n_i+n_o)n_o & \text{if } n_h = 0 \ \& \text{ recurrent} \\ (1+n_i+n_h)n_h + (1+n_h+n_o)n_o & \text{if } n_h > 0 \ \& \text{ recurrent} \end{cases} \tag{1}$$

where n_c, n_i, n_h, n_o and “1” represent the numbers of fully connected weights, inputs, hidden neurons, outputs, and biases, respectively.

When the ANN structure is also evolved, n_h becomes a design variable to be optimized. So the chromosome length must also be variable to accommodate the variable ANN structure and evolve solutions of suitable complexity. To give the algorithm more freedom to search for the appropriate network structures, fewer restrictions are imposed on the number of permissible connections and the variable chromosome length is computed as follows:

$$n_c = \begin{cases} (1+n_i)n_h + (1+n_i+n_h)n_o & \text{if feed-forward} \\ (1+n_i+n_h+n_o)(n_h+n_o) & \text{if recurrent} \end{cases} \tag{2}$$

where n_c represents the maximum possible number of connections, but not all of them must be active. A non-zero real value in the genotype vector represents the weight value of an active connection, while zeros represent inactive (non-existent) connections. Note that the $n_h = 0$ cases of (1) are included in (2).

2.2 Initialization

The population is randomly initialized at the beginning of an evolutionary run. For fixed network structure cases, all the genotype vectors are of the

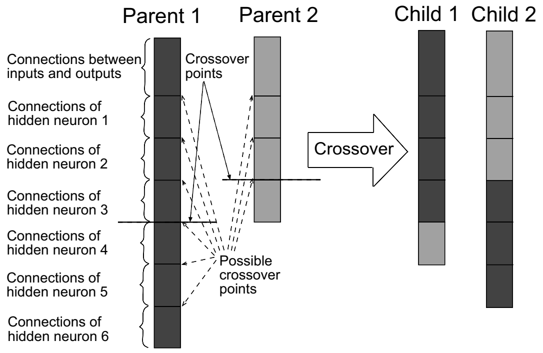

Parent 1
Parent 2
Child 1
Child 2
Connections between inputs and outputs
Connections of hidden neuron 1
Connections of hidden neuron 2
Connections of hidden neuron 3
Connections of hidden neuron 4
Connections of hidden neuron 5
Connections of hidden neuron 6
Crossover points
Crossover
Possible crossover points

Fig. 1. Illustration of crossover scheme for two chromosomes of different lengths

same length with random values. For variable network structure cases, first n_h is randomly selected from 1 to 10[3] for each individual, then the genotype vector length is computed by (2), and each real number in the genotype vector is set to a random value (active) or zero (inactive) with probability 50%.

The networks initialized this way might contain some useless hidden neurons which do not contribute to the outputs at all. To improve the algorithm efficiency, they are identified and removed from the network by a simple routine after initialization to make the network more concise and relevant.

2.3 Genetic Operations

Crossover and mutation are both used in the evolutionary algorithm here.

In fixed ANN structure cases, standard crossover operators can be directly applied to two real vectors of equal length. In variable ANN structure cases, the crossover must operate on two vectors of different lengths which represent two distinct network structures. It is known that evolutions relying on crossover do not perform well in ANN topology optimization because of their intrinsic disruption feature and the permutation problem [18]. To protect possible modules in the network, crossover points can not be arbitrarily chosen along the whole genotype vector. As shown in Fig. 1, the genotype vector can be grouped into sub-vectors or blocks according to the hidden neurons, and the crossover implemented here only allows block-wise interchange between the parents. The algorithm also tries to best match the sequence of the hidden neuron blocks of before crossover to reduce influence of the permutation problem. Then, from all possible crossover points, as shown in Fig. 1, a random crossover point is selected for both parents and their hidden neuron blocks below the crossover point are exchanged to create two new offspring. For the example shown in Fig. 1, two parents networks of 6 and 3 hidden neurons produce two children networks of 4 and 5 hidden neurons respectively.

[3] There is, however, no upper limit for n_h during the evolution.

Mutation is also a powerful tool for creating new network structures and synaptic weights. In fixed structure cases, only Gaussian mutation is used to change values of the synaptic weights. In variable structure cases, two extra types of mutations are introduced. First, a hidden neuron could be added to or removed from the current network configuration. Second, a connection between any two neurons could be turned on or off by switching between non-zero and zero values. When a hidden neuron is added or a connection is switched on, the synaptic weight values are initialized as described in Sect. 2.2.

The crossover and mutation operations could also introduce useless hidden neurons and identical individuals, which are both removed from the network before evaluation. If identical copies are allowed to exist in the population, the power of crossover and pool diversity are reduced, which might cause pre-convergence of the evolution.

3 Case Study: Collective Robotic Inspection

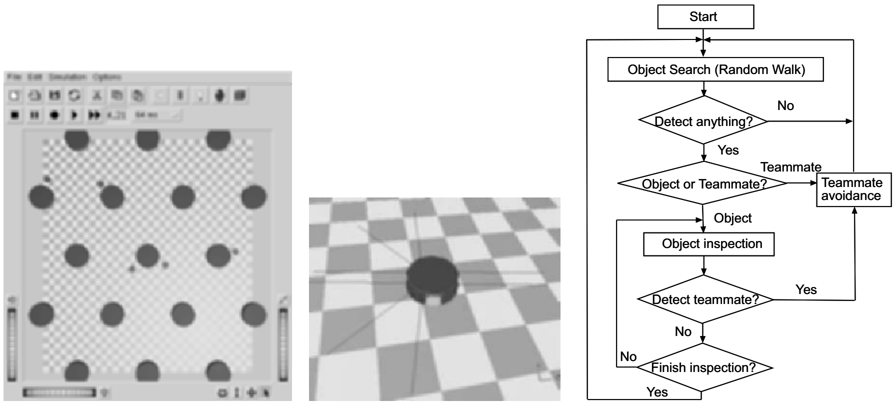
This case study is concerned with the automatic synthesis of ANN-based control algorithms to inspect regular structures using a homogeneous robot swarm. The goal is to design "local" algorithms that coordinate the fine motions of the moving platforms. Sensor uncertainty and vehicle position uncertainty should be taken into account when planning the local motions that carry out the gross motion plan, i.e., the collective effect of the multi-vehicle platform as a whole.

3.1 Application Background

Autonomous robots find a wide variety of applications in the real world to release humans from various chores, especially when the working environment is hazardous or not easily accessible by human beings. For instance, inspection of human occupied space transportation systems and platforms is currently heavily labor intensive, costly and time consuming. Another example could be the inspection of propulsion systems (such as jet turbines), which is usually performed visually by human experts using borescopes, a process which is also time consuming and cost intensive [9]. Therefore it is desirable to have the inspection task performed autonomously by a swarm of miniature mobile robots in these situations. This idea is intellectually appealing and it could find broad applications for general inspection of engineered or natural structures.

3.2 Experiment Setup and Simulation

This paper considers a simple 2-dimensional (2D) scenario, where the objects to be inspected have regular circular shapes (20 cm in diameter), as shown in Fig. 2(a). It is assumed that completely circumnavigating an object is a good



(a) (b) (c)

Fig. 2. (**a**) Top view of structure inspection setup in the embodied simulator: the bigger disks represent the cylindrical objects to be inspected while the smaller dots are the robots; (**b**) Close-up of the robot model with its distance sensor rays (*solid lines*); (**c**) The logical diagram of the rule-based hand-coded controller.

emulation of the scanning-for-flaws maneuver. A continuous world scenario without walls is also simulated by wrapping the robots position around to the opposite side when they move out of one side.

The collective performance measure for this scenario depends on the ratio of the inspected object surfaces over a pre-specified time span to all that needed to be inspected in the world. Therefore the maximum performance “1” can be achieved by complete coverage, i.e., fully inspecting all distinct objects in the world, within the time limit. Note that only 12 distinct objects are present in the world shown in Fig. 2(a) under the wrap-around condition.

The simulation scenario shown in Fig. 2 is implemented in Webots[4] [10], a 3-dimensional (3D), embodied, sensor-based, kinematic simulator. As shown in Fig. 2(b), the simulated robots are miniature unicycle vehicles (5 cm in diameter), like the Khepera[5] robots. They are equipped with eight distance sensors as ANN inputs with extended sensor range (10 cm). The sensors are assumed to be line sensors characterized by a linear response with the distance: the closer the sensed object the higher the value. Integer sensor values ranging from [0, 1023] with ±10 white noise are normalized to [0, 1] before feeding to the ANN controller. Both hidden (if any) and output neurons use sigmoid output functions producing outputs in the range of [0, 1]. The ANN has two outputs mapping to the two wheel speeds[6] of the robots, taking integer values from [-20, 20], with each speed unit representing 8 mm/s in the real world.

[4] `http://www.cyberbotics.com/products/webots/`

[5] `http://www.k-team.com/robots/khepera/`

[6] Either the left and right speeds, or the forward and rotation speeds.

Wheel slippage is also simulated with ±10% white noise on wheel speeds at each simulation step.

The inspection task requires the robot to approach an object to be inspected, inspect it with minimal redundancy and maximum speed, leave it and search for other objects. In collective scenario the robot also needs to avoid teammates. Although one could implement a rule-based hand-coded controller (see Sect. 3.3) or apply behavior-based control algorithms [4], this is nevertheless a non-trivial task for a neural controller reading eight crude distance sensors and controlling two motor wheels directly. Indeed, the controller must not only evolve basic reactive behaviors such as object search, object inspection (i.e., follow the contour of the object) and teammate avoidance but also correctly *sequence* them: for instance, switching from object search to inspection when an object is found and searching for new objects after fully inspecting an object.

3.3 Hand-coded Controller

In order to create a baseline of what level of performance is achievable in this specific case by a traditional design method based on human intelligence, a simple hand-coded controller based on logical rules has been implemented for the same task. It exploits exactly the same sensor inputs and controls the same motor outputs as the evolved ones, and can be used to evaluate and compare with the evolved ANN controller solutions. As shown in Fig. 2(c), the hand-coded controller is based on the robot's distance sensor inputs and some internal timers. There is one key parameter that controls how long the robot keeps inspecting an object. This parameter has been optimized by systematic search in order to get the best performance.

Although it is rather straightforward to implement such a rule-based hand-coded controller for the simple scenario defined in Sect. 3.2, it is not obvious how to complete the same task with a structural ANN controller. Moreover, it might become more intractable and even infeasible to implement such a hand-coded controller for more complex (*e.g.*, inspection of 3D, irregular structures) scenarios, where evolutionary algorithms might be more appealing.

4 Results and Discussions

The evolutionary algorithm was applied to evolve ANN controllers under different configuration settings: feed-forward or recurrent networks with or without a variable number of hidden neurons, as shown in Table 1. The population sizes of the evolutions depend on the dimension of the genotype vector to be optimized. The first two types of neural networks are of fixed simplest topologies with shortest chromosome lengths and smaller pool sizes. The latter two types of neural networks are of variable structures with longer (average) chromosome lengths and larger pool sizes.

Table 1. Different ANN types considered in the evolutionary algorithm

ANN type description	**symbol**	**pool size**
feed-forward without any hidden neurons	ffnh	50
recurrent without any hidden neurons	rcnh	50
feed-forward with n_h^* hidden neurons	ffvh	100
recurrent with n_h^* hidden neurons	rcvh	150

*Note $n_h \geq 0$ is a variable number.

For each type of ANN controller synthesis, a series of evolutionary experiments were conducted with different output speed maps and different coefficients in the sigmoid neuron output function. For each evolutionary experiment, 5 evolutionary runs with different random seeds were performed, each lasting for 200 generations.

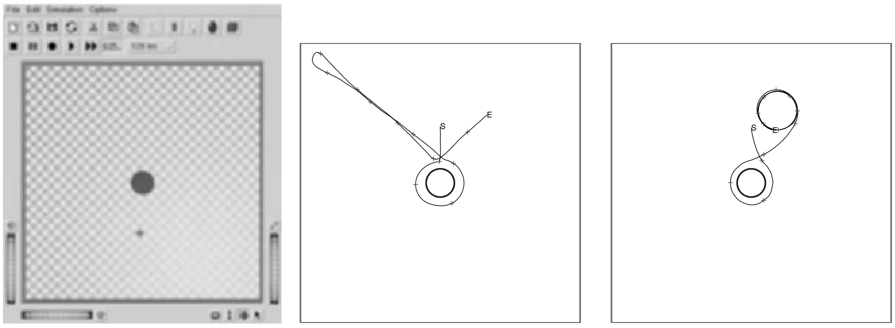
Due to the noise present in the fitness evaluation of controller solutions in the simulated environment, a noise test of 100 evaluations was applied to each individual in the final population of each evolutionary run as well as the hand-coded controller to get fairer performance comparisons of different controllers (refer to Fig. 6). Different aggregation criteria (minimum, geometric mean, average, etc.) can be used as a measure to estimate the overall performance of each solution from its multiple fitness values. However, only one single evaluation was applied for each individual during the evolutions, because the evaluations in the simulated environment are significantly more computationally expensive than the genetic operations and in this case single evaluation (smallest sample size) is the most computationally effective strategy [5].

In the following sections, the inspection task was approached by three systematic steps: the single robot single object scenario (Sect. 4.1), the single robot multiple objects scenario (Sect. 4.2), and the multiple robots multiple objects scenario (Sect. 4.3).

4.1 Single Robot Single Object (SRSO) Scenario

The global problem of collective robotic inspection of multiple objects in the world can be decomposed to the microscopic interaction between a single robot and a single object, as shown in Fig. 3(a). Here the goal of the robot is to make one and *only one* full circle of the object as soon as possible without any collisions. A short evaluation span of 500 time steps[7] was chosen here. The robot always started at the same initial position and orientation (facing the object) for all evaluation spans here to reduce noise effects. Walls were included in this scenario to facilitate the development of object avoidance behavior of the ANN controllers during evolutions. Walls can be distinguished from the circular-shaped object by their different sensory patterns.

[7] Each time step simulates 64 milliseconds in real time.



(a) (b) (c)

Fig. 3. (**a**) Screen shot of SRSO scenario; sample robot trajectories for 500 time steps (32 s) using the hand-coded controller (**b**) and the best evolved controller (**c**): "S" represents the constant starting point and "E" the ending points, with "+" symbols placed along the trajectories every 40 time steps (2.56 s)

Figure 3 shows the sample robot trajectories of the hand-coded controller and the best ANN controller evolved. It is interesting to note the distinct behaviors of the two controllers shown here. The rule-based hand-coded controller clearly follows the logic shown in Fig. 2(c): it goes directly to the object, walks around it, then leaves and starts random walk. While the evolved one just hangs around in circles after it finishes the inspection task.

It turned out that for this scenario neural controllers that had access to an additional timer input achieved better results than those had not, and were comparable to the hand-coded controller, which also used timers. This implies that timing was a key factor here due to the lack of spatial clues in the world.

4.2 Single Robot Multiple Objects (SRMO) Scenario

A single robot is let to explore the multi-object scenario shown in Fig. 2(a) in the SRMO scenario. Again the goal here is to inspect (circle around) as much as possible the 12 distinct circular-shaped objects in the world. No walls were simulated in this scenario since wrap-around was applied to simulate a continuous world, similar to an unfolded ball. The evaluation span was 2000 time steps here for each ANN candidate controller during evolutions. The robot starts from a random initial position and orientation for each evaluation.

Figure 4 shows sample robot trajectories of the hand-coded and evolved controllers in the SRMO scenario. The difference is quite obvious. The hand-coded controller always tries to make a full circle of each object it finds, then walks away in rather straight lines to search for "new" objects[8]. Different evolved controllers show a variety of different behaviors. The one shown in

[8] Note that the robot had no clue to figure out whether a newly discovered object was inspected before or not according to Fig. 2(c).

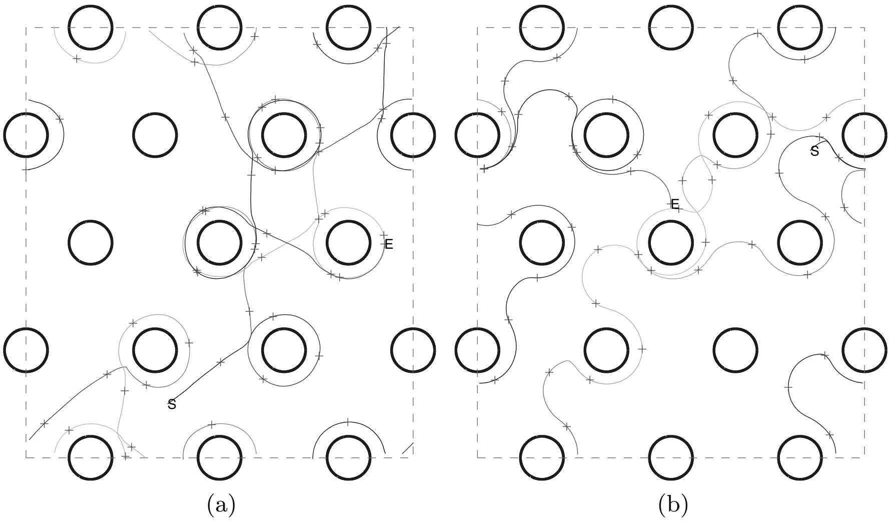


Fig. 4. Sample robot trajectories of the SRMO scenario for 2000 time steps (128 s) using the hand-coded controller (**a**) and the evolved controller (**b**). The dashed lines delimit the wrap-around boundaries; "S" represents the random initial starting points and "E" the ending points. The trajectories are shown in gradually changing colors with "+" symbols every 40 time steps (2.56 s)

Fig. 4(b) always walks in alternate curves: counterclockwise and fitly curved when inspecting an object while clockwise and less curved otherwise. Most engineered solutions would probably apply the strategy of the hand-coded controller naturally, it is surprising to discover that this evolved ANN-based strategy could work equally well here. In addition, the evolved controllers no longer need additional temporal input here to be comparable with the performance of the hand-coded controller, which still depends on its timers.

However, it might be more difficult for the evolved controllers to achieve complete coverage of all objects. Because it might leave (drift away from) an object before fully inspecting it, it would generally take longer to fully inspect all objects in the world than the hand-coded one. One possible cause might be that the robot could hardly achieve complete coverage within the evaluation span, hence there was no pressure in the evolution to favor complete coverage.

4.3 Multiple Robots Multiple Objects (MRMO) Scenario

Finally a homogeneous team of five robots were employed to collectively inspect the multiple (12) circular objects in the world, as shown in Fig. 2(a). The goal as well as the wrap-around and random initial conditions were the same as those in Sect. 4.2, with an evaluation span of 800 time steps.

Figure 5 shows the sample robot trajectories of the hand-coded and evolved controller for the MRMO scenario. Both behaviors seem to follow the respec-

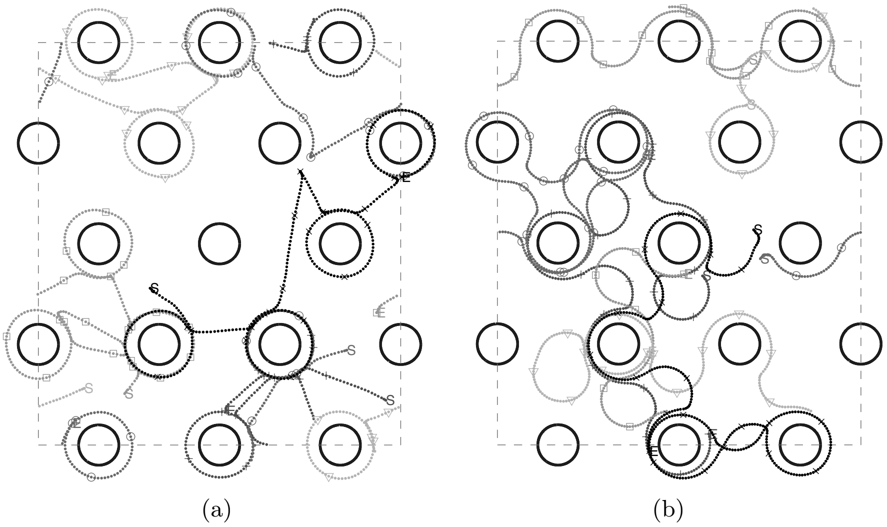
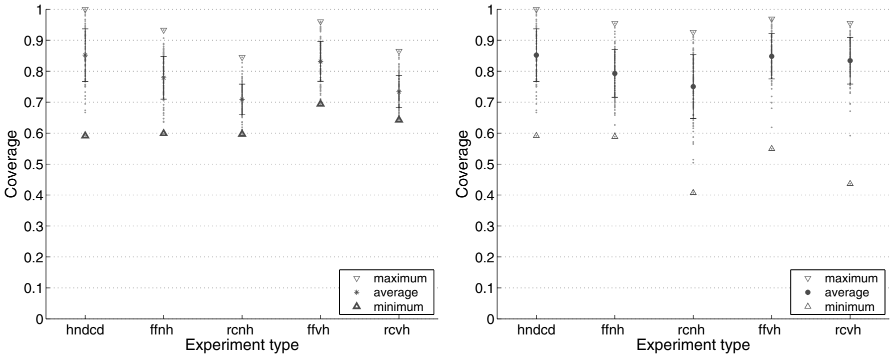


Fig. 5. Sample robot trajectories of the MRMO scenario for 800 time steps (51.2 s) using the hand-coded controller (**a**) and the evolved controller (**b**). The dashed lines delimit the wrap-around boundaries, but some trajectories beyond the boundaries were kept to enhance the display. "S" represents the random initial starting points and "E" the ending points. Different robots' trajectories are shown in different colors with different markers placed every 40 time steps (2.56 s)

tive strategies discussed in Sect. 4.2. It is noteworthy for the evolution to develop the obstacle (including teammates) avoidance behavior, especially when it was not explicitly defined in the fitness function.

Figure 6 shows the performances of the hand-coded and best evolved controllers under different ANN types (as shown in Table 1) for the MRMO scenario. It is shown that the evolved controllers, especially those with a variable ANN topology, seem to have achieved comparable performances as the hand-coded controller in terms of average performance, and even slightly beat the hand-coded controller in terms of worst performance, appearing more robust to noise. It is remarkable for the evolutionary algorithms to automatically discover robust solutions from only single noisy evaluation of each candidate solution during the evolutions, which verified its extraordinary ability to work in noisy environments.

It is also observed that controllers evolved with variable ANN topologies can generally achieve better results than those with fixed ANN topologies. This demonstrated the power of evolutionary algorithms to synthesize appropriate ANN topologies for a given problem, and evolve the necessary synaptic weights simultaneously. Although the best evolved control strategies have achieve the same level of performance as each other as well as the hand-coded controller, their underlying ANN topologies are completely different, including



(a) (b)

Fig. 6. Coverage values achieved by the hand-coded controller (hndcd) and the best controllers evolved under different ANN architectures (refer to the symbols in Table 1) and selected according to (**a**) minimum and (**b**) average performance for the MRMO scenario shown in Fig. 2(a). Each column shows the coverage values (*the green dots*) obtained by one controller during the 100 evaluations in its noise test and the error bars indicate the standard deviation

both feed-forward and recurrent ANN's with a number of hidden neurons from two to six. They can provide human engineers with diverse alternative candidate solutions that might be difficult to conceive from human intelligence. On the other hand, no significant performance differences were observed between the feed-forward and recurrent ANN's in all scenarios when all other conditions were the same.

It is also noted that the evolutionary algorithm is able to adapt the controller solutions according to the collective or single robot scenarios. Hence the controllers evolved in collective scenarios can achieve better results in collective scenarios than those evolved in single robot scenarios, and vice versa.

5 Conclusion and Future Work

An evolutionary algorithm was applied to automatically synthesize neural controllers for autonomous robots in a noisy simulation environment with little prior knowledge on ANN topologies. The methodology was validated in the framework of a case study concerned with collective robotic inspection of 2D regular structures. It was shown that the best evolved controllers can achieve excellent and robust performances with a variety of different ANN architectures, providing multiple good candidate solutions for human engineers.

In the future, the same synthesis methodology will be applied to more complex and realistic problems such as collective robotic inspection of 3D irregular space structures and/or jet propulsion systems. Implementation and verification of evolved controllers with real robots would also be meaningful.

Acknowledgments

This material is based upon work supported, in part, by NASA Glenn Research Center and by the Engineering Research Centers Program of the National Science Foundation under Award Number EEC-9402726. Martinoli A is currently sponsored by a Swiss NSF professorship.

References

1. Antonsson EK, Zhang Y, Martinoli A (2003) Evolving engineering design trade-offs. In Proc. 15th Int. Conf. on Design Theory and Methodology, ASME. Paper no. DETC2003/DTM-48676.
2. Bäck T (1996) Evolutionary algorithms in theory and practice. Oxford University Press, New York, NY.
3. Bonabeau E, Dorigo M, Theraulaz G (1999) Swarm intelligence: From natural to artificial systems. Oxford University Press, New York, NY.
4. Correll N, Martinoli A (2004) Modeling and optimization of a swarm-intelligent inspection system. In Proc. 7th Symp. on Distributed Autonomous Robotic System (DARS), Springer-Verlag.
5. Fitzpatrick JM, Grefenstette JJ (1988) Genetic algorithms in noisy environments. Machine Learning, **3**(2-3):101–120.
6. Goldberg DE (1989) Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, Reading, MA.
7. Hayes AT, Martinoli A, Goodman RM (2003) Swarm robotic odor localization: Off-line optimization and validation with real robots. Robotica, **21**(4):427–441.
8. Hertz J, Krogh A, Palmer RG (1991) Introduction to the theory of neural computation. Perseus Books, Reading, MA.
9. Martin K, Stewart C (2000) Real time tracking of borescope tip pose. Image and Vision Computing, **10**(18):795–804.
10. Michel O (2004) Webots: Professional mobile robot simulation. Journal of Advanced Robotic Systems, **1**(1):39–42.
11. Miglino O, Lund HH, Nolfi S (1995) Evolving mobile robots in simulated and real environments. Artificial Life, **2**(4):417–434.
12. Mitchell M (1996) An introduction to genetic algorithms. The MIT Press, Cambridge, MA.
13. Nolfi S, Floreano D (2000) Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines. The MIT Press, Cambridge, MA.
14. Nolfi S, Parisi D (2002) Evolution of artificial neural networks. In Handbook of brain theory and neural networks, MIT Press, pp. 418–421.
15. Patel M, Honavar V, Balakrishnan K Eds. (2001) Advances in the evolutionary synthesis of intelligent agents. The MIT Press, Cambridge, MA.
16. Scott MJ, Antonsson EK (1998) Aggregation functions for engineering design trade-offs. Fuzzy Sets and Systems, **99**(3):253–264.
17. Versino C, Gambardella LM (1997) Learning real team solutions. In Distributed Artificial Intelligence Meets Machine Learning, Springer-Verlag, pp. 40–61.
18. Yao X (1999) Evolving artificial neural networks. Proceedings of the IEEE, **87**(9):1423–1447.

A Self-Contained Traversability Sensor for Safe Mobile Robot Guidance in Unknown Terrain

Ayanna Howard, Edward Tunstel

NASA Jet Propulsion Laboratory, California Institute of Technology
Pasadena, CA 91109, USA

Abstract. Autonomous mobile robots capable of intelligent behavior must operate with minimal human interaction, have the capability to utilize local resources, and routinely make closed-loop decisions in real-time based on sensor data inputs. One of the bottlenecks in achieving this is an often computationally intensive perception process. In this paper, we discuss a class of cognitive sensor devices capable of intelligent perception that can facilitate intelligent behavior. The primary emphasis is on achieving safe mobile guidance for planetary exploration by distributing some of the perception functionality to self-contained sensors. An example cognitive sensor, called the traversability sensor, is presented, which consists of a camera and embedded processor coupled with an intelligent visual perception algorithm. The sensor determines local terrain traversability in natural outdoor environments and, accordingly, directs movement of a mobile robot toward the safest visible terrain area in a self-contained fashion, placing minimal burden on the main processor. A cognitive sensor design approach is presented and a traversability sensor prototype is described.

Keywords: mobile robot, rover, traversability, cognitive sensor, fuzzy logic, distributed processing

1 Introduction

For planetary exploration missions, there is a necessity to build navigation sensors that enable mobile robot operation in unknown environments with little human intervention. The sensors must possess the ability to acquire new information and utilize on-board knowledge and information processing so that key decisions can be made in real-time. Currently, in real-world robotics applications, this sensor capability requires the availability of powerful on-board computing power and complex interfaces in order to main function of each sensor is to retrieve data for a specific purpose. For example, for planetary exploration missions, the development of a sensor capable of informing mission specialists when water has been detected, in addition to transmitting science data to human specialists for future analysis, is highly desired over a sensor whose main capability is only the latter function of data transmission. To this effect, sensor devices are desired to facilitate intelligent behavior by robots that must operate with minimal ground support (from Earth-based mission control), be capable of utilizing local resources, and routinely make closed-loop decisions in real-time based on sensor data (Noor et al. 2000). One way to advance sensor capability and achieve intelligent perception is to build processing and decision making functionality directly into sensors to make them smart and more cognitive. This approach is adopted herein and applied to a prototype cognitive sensor for mobile robot guidance.

The term "smart sensor" was first coined in the mid-1980's by (Giachino 1986). Since then, many descriptions for smart sensors have arisen (Travis 1995, Dierauer 1998). To offset some of the confusion, the IEEE and U.S. National Institute of Standards and Technology (NIST) derived the smart-sensor standard, which standardizes the networking protocol and features embedded in sensor technology (Frank 2000). Overall, the prevailing definition of a smart sensor classifies such a device as a sensor that incorporates communication capability, self-diagnostics, and intelligence (decision making). Of course, utilizing such a broad criteria for smartness ensures that the class of smart sensors encompasses a wide range of variations. For example, intelligence could represent anything from simple switching logic to a complicated sequence of behaviors. Thus, in order to realistically categorize smart devices, a classification or taxonomy must be utilized to discriminate between the extremes of intelligence.

In Section 2, we propose a taxonomy for smart sensor classes based on levels of intelligence. We introduce the notion of a cognitive sensor in Section 3 and an associated design approach. We demonstrate and apply this approach in Section 4 with a focus on a cognitive sensor for robot guidance that utilizes computer vision and fuzzy logic reasoning to achieve intelligent perception. Section 5 presents preliminary results showing that the sensor maps image data directly to safe motion directives useful for mobile robot guidance. Section 6 provides a brief discussion of related sensing alternatives and section 7 concludes the paper and summarizes some of the key advantages.

2 Intelligence Hierarchy for Sensor Devices

By analyzing the intelligence capabilities that advanced sensor devices should exhibit, five main attributes are derived. Advanced sensors must be capable of:

□ *Self-knowledge*: identifying its purpose and understanding its operational functions
□ *Communication*: transmitting/receiving information (versus raw data) to/from other devices
□ *Perception*: the ability to recognize, interpret, and understand sensory stimuli
□ *Reasoning*: making decisions based on perception of sensory stimuli
□ *Cognition*: the intellectual process of knowing, which includes aspects such as awareness, perception, reasoning, and judgment.

Based on these desired capabilities, a hierarchy is defined for grouping sensor devices together based on their intelligence. This process provides us with three layers of sensor devices: smart sensors, intelligent sensors, and cognitive sensors (Fig. 1). A smart sensor exhibits attributes of self-knowledge and communication, such as defined by (Giachino 1986). An intelligent sensor builds upon the smart sensor construct, and incorporates the perception trait into its feature set. The cognitive sensor further expands its capabilities by including the qualities of reasoning and cognition into its design.

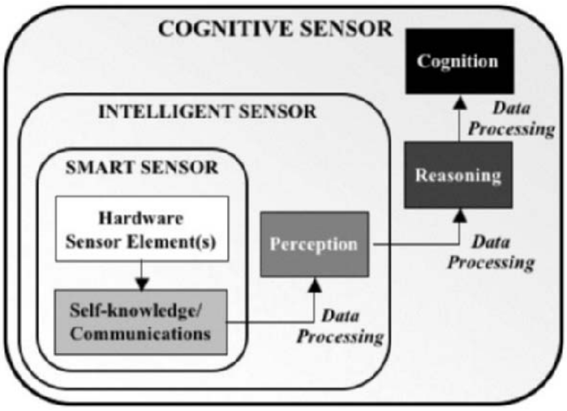


Fig. 1. Intelligence hierarchy for sensor devices.

To truly understand this hierarchy, consider the following example of a sensor capable of intelligent fault tolerance in a robotic system (Fig. 2). In this case, a sensor capable of fault detection would be classified as a smart sensor. It possesses self-knowledge, for it can detect anomalies based on its standard operation, but it does not "understand" the cause of an anomaly and thus would not be classified as an intelligent, or cognitive sensor. On the other hand, a sensor capable of fault diagnosis not only detects faults as they occur, but can determine the cause of an anomaly by interpreting the data input in real-time. This type of sensor would be classified as an intelligent sensor. Continuing with this logic, a cognitive sensor would be represented by a sensor capable of fault repair, in addition to detection and diagnosis. Such a sensor would not only reason about the data, but would be able to decide how to overcome its deficiencies by using its embedded knowledge base. Cognitive sensors may implicitly include world modeling as a function of cognition. While this is not a requirement, they typically include some form of internal representation contrived from raw sensor inputs that can be reasoned about to achieve a concept of "knowing."

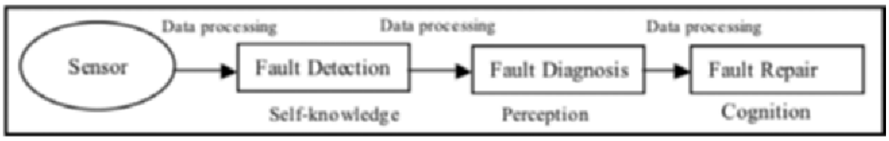


Fig. 2. Intelligence transition for hypothetical spacecraft fault tolerance sensor.

3 Cognitive Sensor Technology

For our immediate purposes, we are mainly concerned with the development of the cognitive sensor, with the realization that the hardware for each sensor type may remain the same whether cognitive, intelligent, or smart. The main difference between each sensor type lies with the intelligence, and thus the focus is on the *software* that is embedded within the device. The physical realization of a cognitive sensor is a hardware sensor device coupled with software-based intelligence embedded in an on-board microcontroller, digital signal processor (DSP), field programmable gate array (FPGA), application-specific integrated circuit (ASIC), or other computing device(s). The hardware structure defines the sensor class, while the software characterizes the type of intelligence capabilities exhibited by the hardware. Our concern is with the development of cognitive sensors for robotic planetary exploration. We thus, further subdivide cognitive sensors for exploration into four main categories:

- *Sampling sensor*: Makes decisions based on detection of static surface specimens in real-time (e.g., detection of water signatures, terrain traversability, surface characteristics)
- *Event-driven sensor*: Makes decisions based on detection of anomalous or impulsive events in novel environments (e.g., Aurora, supernova, meteor showers)
- *Environmental sensor*: Makes decisions based on perception of changes in environmental conditions (e.g., dust storms, flooding, seismic activity)
- *Science sensor*: Makes decisions based on scientific knowledge derived through learning and reasoning about the environment (e.g. science-driven sample analysis, scientific discovery)

3.1 Cognitive Sensor Design Process

The development of a cognitive sensor involves embedding hardware and software into one unified cognitive system. A general design process begins with identification and selection of sensing elements or devices based on the range of desired applications. Requirements derived from the functional description of a particular application are used to drive the refinement of sensor device specifications, embedded software design, and any necessary hardware augmentations (e.g., actuator selection for active sensing systems). Following the selection of baseline components of the cognitive sensor, interfacing and data flow issues must be addressed. The interplay of architectural interfaces defining data flow between the key components and the processor, as well as sensing and control signals among them must be carefully considered.

Specification of a baseline sensor architecture is followed by consideration of operational trade-offs. A cognitive sensor may be thought of as a distributed sensing and computing system. A common design issue for such systems is the determination of the appropriate distribution of intelligence amongst the system components. The cognitive sensor could be comprised of very simple sensor elements coupled with a powerful core processor or software application that functions as the primary source of intelligence. Alternatively, the cognitive sensor may be comprised of smart or intelligent sensors coupled with a very modest processor or software application, in which case the intelligence resides primarily in the sensing elements. An appropriate trade-off between these extremes must be made based on consideration of the target application and the capabilities of selected components. At this point in the design process, the cognitive sensor is simply a programmable sensing and computing device. The design process may conclude with a typical embedded software design and development cycle focused on the target application. With a more or less static hardware architectural configuration, a range of capabilities for a given cognitive sensor may be realized by simply changing the embedded software application — a means to achieve reusability of the same underlying hardware infrastructure for alternative sensing functionality.

3.2 Motivation and Aims

To demonstrate the concepts discussed thus far, we set out to design a cognitive sensor for mobile robot guidance with a focus on sensing terrain traversability characteristics — useful knowledge for augmenting the navigation capability of a planetary mobile robot, or rover. The expected benefit is that a cognitive traversability sensor would increase the intelligence of rovers and allow efficiency of implementations beyond existing navigation perception solutions (or that supplement them). We maintain that this could be achieved if a significant portion of perceptual processing, intelligent reasoning, and overall complexity could be off-loaded from the main processor. Such savings and improvements are desired because in planetary science missions, advanced sensing for engineering operations is secondary to fundamental science data acquisition. Therefore, it is desirable to allocate the most computational resources for the latter. Providing an alternative source of intelligent processing to assist in operations thus allows a corresponding increase in rover autonomy, without sacrificing the primary goal of the mission (increased science return).

Since science is the primary objective of planetary surface missions, significant percentages of rover resources, mass, and volume are dedicated to onboard science instruments (versus navigation sensors). Vision sensors are most commonly used to perceive the environment for navigation. Passive stereovision is preferred over some of the alternative active ranging sensors for power related reasons. Since it is passive, sunlight provides the required energy for daylight operations. Hence, minimal power is required by the passive stereo electronics to serve the dual purpose of providing sensing for navigation and scientific knowledge of the environment through images (Goldberg 2002). Low power consumption by navigation sensors (and other systems) makes way for increased science throughput; and sensor systems of low mass permit larger payloads for science instruments. Advances are required for stereo machine vision to reduce the computational complexity of related algorithms, to decrease the memory required, and/or to increase the computational speed of radiation-hardened spacecraft/rover processors (the latest rovers operated on Mars used 25 MHz processors). For recent insight into the state-of-the-art for dealing with rover stereo image processing complexity see (Goldberg 2002). We seek to lighten the main processor resource (computation, memory, etc) consumption and also increase the intelligence of the sensing/perception functionality. These objectives are achieved in this work by distributing the computing function for sensing to the sensor itself and embedding computational intelligence within the sensor's dedicated resources. In the next section we develop a prototype traversability sensor employing a simple hardware architecture and intelligent reasoning about terrain traversability based on fuzzy logic, all embedded in a cognitive sensing framework.

4 Traversability Sensor Design

We desire a traversability sensor that determines local terrain traversability in natural outdoor environments and, accordingly, directs movement of a mobile robot toward the safest visible terrain area. Terrain traversability quantifies the ease-of-traversal of the terrain by the mobile robot and is used to classify terrain characteristics that directly account for robot traversal difficulty (Howard and Seraji 2001).

The first step in designing the traversability sensor begins by identifying the sensing elements needed to fulfill the functional task requirements. In this case, the traversability sensor is required to image the 180° area located in front of the mobile robot (extending out to 3-5 meters), determine traversability constraints, and output necessary turn angles to command the mobile robot to face the safest region for traversal (Fig. 3). The baseline components identified for prototype sensor development include a CMOS digital camera capable of imaging at 80x60 pixel array size, a 32-bit microcontroller running at 25 MHz, and a simple servomotor for camera pan actuation, all powered by a 9 volt battery. The microcontroller used in the initial prototype is an EyeBot controller developed for embedded mobile robot control and vision systems (Bräunl, 2004). Note that while this controller was a convenient choice for rapid development of the initial prototype, it is possible that the sensor could be realized using less

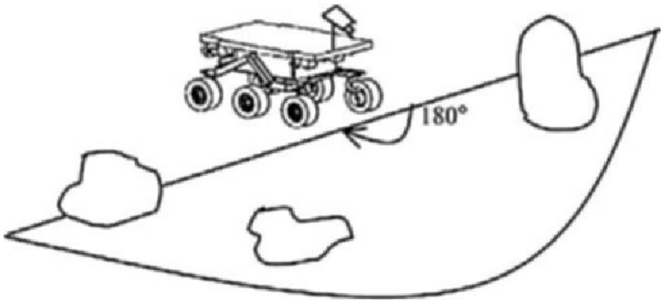


Fig. 3. Imaging the terrain for traversability analysis.

elaborate computing resources. For example, a microcontroller that is less capable in terms of bit resolution and processor speed may be sufficient, and for end use, there may be advantages to minimal realizations in DSP, FPGA, or ASIC components.

For the prototype, interfacing and data flow are enabled by a serial connection between the camera and embedded microcontroller in order to acquire image data for intelligent onboard processing. Also, since the cognitive sensor device is capable of processing the images without transmitting frames to the main processor of the host robotic system, the connection between the sensor and robot control system only requires the transmission of a small number of information bits to relay turn angle (represented by an integer value, which corresponds to 2 bytes in our processing system). Thus, we utilize a simple serial connection to feed the traversability sensor output to the mobile robot to safely guide its motion during navigation. Fig. 4 shows a simplified block diagram of the traversability sensor architecture and a picture of the prototype device. Having established a baseline hardware architecture for the desired traversability sensor we must now couple it with an intelligent software algorithm to be embedded with the microcontroller.

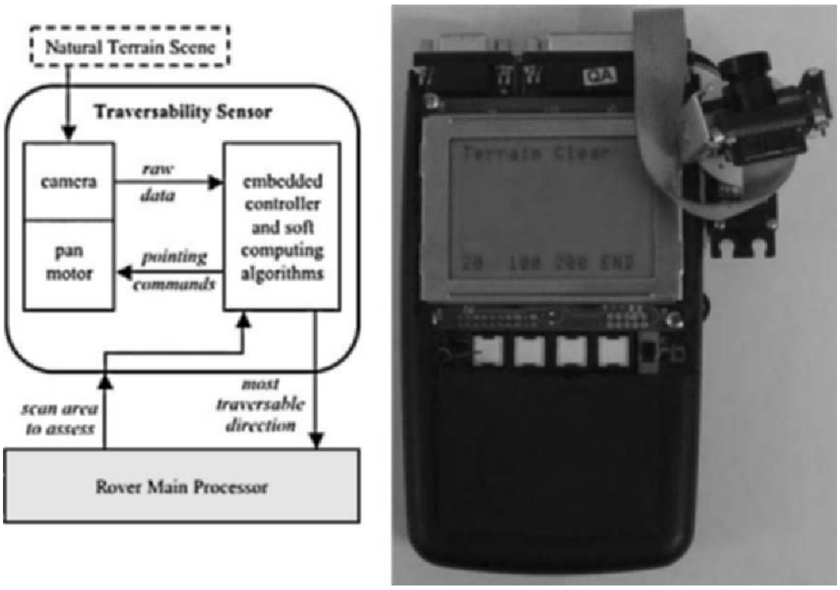


Fig. 4. Cognitive traversability sensor architecture and preliminary prototype.

4.1 Intelligent Software for Terrain Assessment

The intent for this application is for the cognitive sensor to assess the terrain traversability for a planetary rover and determine the safest traversable region for rover navigation. In this case, we have concentrated on embedding an algorithm that determines regional traversability based on rock distribution. The original intelligent visual perception algorithm utilized for this application was developed in (Howard and Seraji 2001). A learning technique was also applied to achieve traversability assessment performance closely resembling that of a human expert in (Howard et al, 2001). Based on visual imagery, our approach determines roughness of the terrain surface and derives traversability quality from this analysis. Terrain roughness refers to the coarseness and surface irregularity of the ground plane to be traversed and directly accounts for rover navigation difficulty based on robot kinematic constraints represented in the software. Additional terrain characteristics that contribute to surface traversal difficulty by a given rover, such as slope and discontinuities, have been considered previously (Howard and Seraji 2001). We have also explored neural network classification of surface types based on image texture analysis for fuzzy control of rover traction on changing terrains (Tunstel and Howard 2003). To illustrate the cognitive sensor concept for the prototype traversability sensor, we restrict the traversability assessment to terrain roughness only. Thus, a portion of the algorithms developed and tested on robots over the last several years is ported to a self-contained embedded solution.

4.1.1 Terrain Image Processing

Visual perception of rock distribution is the basis for assessing terrain surface roughness. First, an algorithm for determining the size and concentration of rocks in a viewable scene is applied to images taken from the robot's vantage point. Rock size and concentration are represented by two parameters: *concentration of small-sized rocks* and *concentration of large-sized rocks*. To identify rock objects, a region-growing method (Horn 1986) consisting of edge detection, ground signature extraction, and obstacle (large rocks/boulders and large groups of rocks) identification is applied to the image. The edge detection algorithm is utilized to segment the image into identifiable regions. Using a one-dimensional operator, called the Canny operator (Horn 1986), edges are extracted and linked together using a search method in which candidate edges are chosen based on proximity. Rocks are identified by extracting objects that have closed contours and comparing with the ground signature using average pixel color and standard deviation values.

To determine the number of small and large-sized rocks contained within the image, the numbers of pixels that comprise target objects are calculated. Those targets with a pixel count less than a user-defined threshold are labeled as belonging to the class of small rocks and those with a count above the threshold are classified as large rocks. All such labeled target objects are then counted in order to determine the small and large rock concentration parameters.

4.1.2 Fuzzy Logic Reasoning

Small and large rock concentrations as well as the concepts of terrain surface roughness and traversability are subjective attributes that can be represented using fuzzy sets and reasoned about using fuzzy logic. Fuzzy logic (Passino and Yurkovich 1998) provides a flexible tool for modeling the relationship between input and output information, and is distinguished by its robustness with respect to noise and variations in system parameters. Linguistic fuzzy sets and conditional statements allow fuzzy systems to reason and make decisions based on imprecise and uncertain information. We exploit these capabilities in this application to effectively deal with the imprecise information extracted from the visual sensors and the uncertainty in the knowledge about the environment.

The range of rock concentrations is partitioned using fuzzy sets with linguistic labels {*FEW*, *MANY*}. The terrain roughness is represented by four fuzzy sets with linguistic labels *{SMOOTH, ROUGH, ROCKY}*. Terrain traversability for safe robotic exploration is represented by fuzzy sets with linguistic labels {*UNSAFE, RISKY, SAFE*}. Thus, a region of unsafe traversability is very difficult for the rover mobility system to traverse, while a highly traversable region is safe and relatively easy to traverse. Fuzzy set membership functions for these terrain characteristics are shown in Fig. 5 (where a, b, and c, can have different values for small and large rock concentrations).

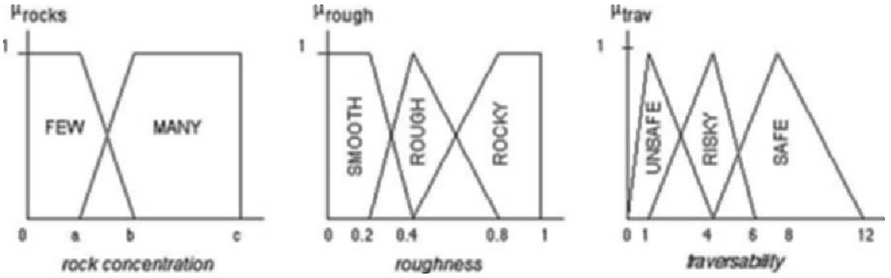


Fig. 5. Membership functions: rock concentration, roughness, and traversability.

Terrain roughness is inferred directly from the rock size and concentration parameters, and the associated traversability is inferred from terrain roughness, according to the rules below. Recall that in more comprehensive implementations additional terrain attributes such as slope and discontinuities would be factored into the rules that infer traversability. In either case, Mamdani rules and max-product inference are employed.

IF SR is *FEW*	AND	LR is *FEW*,	THEN	Roughness is *SMOOTH*; Traversability is *SAFE*
IF SR is *MANY*	AND	LR is *FEW*,	THEN	Roughness is *ROUGH*; Traversability is *RISKY*
IF LR is *MANY*			THEN	Roughness is *ROCKY*; Traversability is *UNSAFE*.

images for traversability assessment. In each period, the rover navigation module would command the traversability sensor to pan the camera and acquire images throughout a specified range of radial coverage. The terrain roughness and traversability in the viewable scene of each image would be classified using the rules above. Once the cognitive sensor reasons about the terrain traversability associated with all regions throughout a maximum 180° viewable terrain area (see Fig. 3), the microcontroller commands the camera to face the safest region. The associated turn angle is then transmitted to the robot motion control system to execute traversal in that direction. This turn angle is typically considered among other turn angles recommended by several motion behaviors from the navigation system. The actual motion executed by the rover is an aggregated function of the recommended motions influenced by the traversability sensor output.

Given this combined hardware-software design, the attributes of the traversability sensor that qualify it as a cognitive sensor device are documented in Table 1, which links them to respective sensor capabilities. This sensor belongs to the sampling sensor category of cognitive sensors for exploration defined in Section 3.

Table 1. Traversability sensor capabilities with attributes of sensor intelligence.

Cognitive Traversability Sensor	
Attributes	Capabilities
Self-knowledge	Fulfills requirements by design through awareness of functional capabilities.
Communication	Directly communicates with robot control system through serial connections.
Perception	Recognizes terrain hazards based on visual sensing capability.
Reasoning	Utilizes derived terrain information to reason about safety for robot traversal.
Cognition	Uses knowledge of terrain hazards and robot capability to employ fuzzy logic in order to emulate human-like judgment and decision-making.

5 Experimental Validation

We briefly present representative examples that validate the intended function of the traversability sensor. The sensor is operated in a stand-alone configuration for these preliminary tests; it is not yet integrated with a mobile platform. Since the output is a turn angle that can be interfaced with a mobile platform's motion control system in a straightforward manner, and since the correctness of pointing to traversable areas can be assessed by a human observer, it is not necessary to validate the sensor in a fully integrated configuration. The sensor performance is independent of a rover's capability to execute the commanded turn motion.

A series of natural terrain images was used as input to the traversability sensor to exercise to computational process of detecting rock sizes and concentrations, inferring viewable terrain roughness and traversability, and generating a recommended rover turn angle as a safe traversal direction. The correctness of each turn angle output of the sensor was judged by the experimenters based on its plausibility given inspection of the images with the human eye. Fig. 6 shows the results for three test images. The original terrain images are shown on the left. The right column of images shows the image processing results of rock detection; these are annotated with the traversability assessment result (linguistic labels). In addition, Fig. 6 includes arrows overlain on the original images indicating the recommended direction for robot traversal. The arrows are oriented relative to a coordinate system in which the 0° azimuth reference angle is aligned with the extreme left side of the image, the 90°angle is straight ahead in the image, and the 180° angle is aligned with the extreme right side of the image. The traversability sensor turn angle outputs relative to the same azimuth reference for these images are roughly: 10° (top image); 102° (middle image); 90° (bottom image).

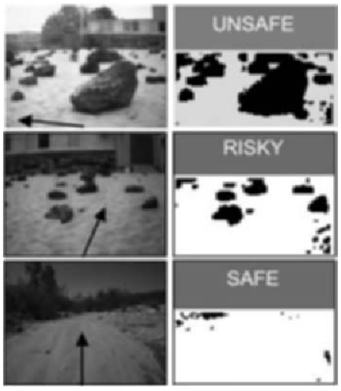


Fig. 6. Representative results of traversability sensor processing and outputs.

In all preliminary tests using these and other images, the traversability sensor processes images and properly generates a safe motion directive for robot execution. The test images of real natural terrain scenes do not include steep slopes or terrain discontinuities such as abrupt drop-offs (cliffs) or elongated gaps (crevasses) since embedded software design is only cognizant of terrain roughness as defined herein. Clearly, this implementation of the sensor will not account for other such terrain hazards to rover mobility. Integration of additional soft computing with the sensor is planned for future development. With this more complete intelligence the traversability sensor would be a viable alternative to existing perception solutions for terrain-based robot navigation guidance. While the traversability sensor and a conventional stereo vision system both produce a guidance directive for rover motion, admittedly, we do not get valuable 3-D local terrain maps that are by-products of stereo processing with this sensor option. This is not to say, however, that cognitive sensors with embedded stereo processing would not be feasible to develop.

6 Discussion

For outdoor mobile robotics applications, a variety of sensors are commonly employed to measure the presence of or range to obstacles, robot bearing and location with respect to the environment, and robot motions. The existing sensor types are as varied as the physical phenomena underlying their operations (e.g., optical, acoustic, inertial, magnetic, etc.). Despite the large variety of available sensor technologies, planetary rover systems cannot readily utilize them due to limitations and/or constraints related to mass, volume, power, and operability/survivability in the space and planetary environments (often non-issues for Earth-based robot applications). The variety of terrain perception and ranging sensors that are viable candidates for robust outdoor mobile robot navigation include: stereo and monocular cameras for computer vision; laser, millimeter-wave, and/or ultrasonic range-finding devices; etc. It is generally not very difficult to select sensors for world perception from among these types that satisfy requirements of general outdoor robotics applications (barring cost as a constraint). However, the requirements for sensors that can be feasibly integrated on planetary rovers differ for those amenable to Earth-based robotics, thus adding a unique twist to an otherwise complicated problem (Tunstel and Howard, 2002). Certain sensor designs and electronic or mechanical characteristics are preferred for selection over others.

The traversability sensor described above is not a direct competitor of other visual sensors or approaches used for outdoor robot navigation. It is an example of a cognitive sensor, which is a new sensor paradigm. As such, direct benchmarking or other comparisons are not very meaningful. Despite this, some of the important concerns about lesser feasible characteristics of related sensors commonly used on Earth-based outdoor robots are worthy of discussion.

For terrain hazard detection sensor systems (e.g., passive stereo vision or active laser rangefinders) solid-state solutions are preferred over systems that require mechanical scanning for area coverage (e.g., laser scanners) (Matthies 1997). If mechanical components must be employed to actuate sensor devices, designs requiring few or no moving parts are preferred. Such components stand a better chance of surviving the vibration and gravity forces of spacecraft launch, atmospheric entry, and landing associated with flights from Earth to other planet surfaces. A common realization for range sensors is based on time-of-flight measurement of acoustic waves. Ultrasonic rangefinders are quite common for indoor robotics applications and to a lesser degree outdoors. They are even less frequently proposed for planetary surface robots. While acoustic sensors will not operate in the vacuum of space, they can be applied on the surface of Mars if the mechanism responsible for generating acoustic waves is effective in its thin, ~95% CO_2 atmosphere. A range sensor of similar popularity is the laser rangefinder. Laser ranging systems extract range information from an image scene, typically using one of two main techniques: active laser scanning and active triangulation (Hebert 2000). We have already alluded to some drawbacks of mechanically scanned devices and power related advantages of passive versus active ranging sensors. These drawbacks aside, passive stereo or other ranging systems available on existing rovers cannot detect certain terrain hazards such as pits filled with loose drift material that may have insufficient load-bearing strength to support a rover. Such hazards are known to exist on the Martian surface (Moore 1999) and need to be detected before a rover engages them. Without additional special-purpose sensors and supporting software algorithms, typical hazard detection and ranging sensors are useless in this regard. Downward-looking impulse radar has been suggested as a proximity sensor for this problem (Speissbach 1988). Similar millimeter-wave and microwave ranging sensors are useful for general outdoor collision avoidance in environments subject to dust, blowing sand, and other all-weather conditions (Everett 1995, Clark and Durrant-Whyte 1997). However, the mass of available radar units may be prohibitive for some planetary rover applications, and candidate radars for terrain property sensing must be low-power and low-mass devices.

These concerns reveal the need for innovative perception systems that comply with complex combinations of constraints on physical design, environmental survivability, and onboard resources of planetary rovers. The viable alternative sensors for perception briefly discussed above perform the basic sensing function and provide raw data to be processed, typically by a robot's central processor. Motion commands for the robot are eventually produced after substantial computational effort expended by the processor; the computational effort, therefore, is not as available for science data acquisition, processing or transmission. Such sensors are thus classified as smart or intelligent at best. On the other hand, the cognitive traversability sensor assumes all or most of the end-to-end function of sensing, perceiving, and reasoning to produce motion commands as its output with no, or minimal, burden on the rover's main processor.

7 Conclusions

The development of the traversability sensor prototype allows us to validate the cognitive sensor structure and provide a concrete example of its utility for real-world applications. In the case of the prototype traversability sensor described herein, since processing of visual data can cause large computational strain on the robot's main processor, embedding the vision-based traversability algorithm within a separate cognitive sensor device facilitates the development of a more efficient real-time robotic system capable of intelligent behavior and safe navigation in unknown terrain.

In contrast to many Earth-based outdoor mobile robot applications, sensor devices and systems employed by planetary mobile robots must comply with hard constraints on mass, power, mechanical complexity, and electrical characteristics. Commonly employed sensing solutions for Earth-based robots are of limited use in other planetary surface environments due to these constraints. Furthermore, existing solutions for planetary robot sensing and perception require improvements that enhance capabilities while increasing efficiency and reducing complexity. Advantages offered by the cognitive traversability sensor relative to conventional perception systems include:

□ use of a single camera use supported by simpler algorithmic processing (relative to the computational burden of stereo processing) for terrain assessment;

□ main processor off-loading, eliminating the need for it to map images to motion directives; and

- embedded human-like reasoning to achieve higher level processing for vision-based perception.

Future directions of this work are aimed at full integration of the cognitive sensor on a mobile robot and documentation of the various design trade-offs necessary for real-world implementation. Plans include enhancing the sensor soft computing capability by integrating the complete algorithm (Howard and Seraji 2001, Tunstel and Howard 2003), which detects and reasons about terrain slope, discontinuities, and surface type in addition to roughness. This would be followed by comparison of the traversability sensor discussed in this paper with a "less-intelligent" sensor configuration based on previous implementations of the same algorithms in which the intelligence resided and executed on the robot's main processor instead of on a self-contained embedded processor. From such comparisons, we would expect a definite computational time-savings in the least. Beyond that, we would expect to yield a basis for future comparison of overall navigation performance on a common mobile robot involving execution of motion directives generated by both the traversability sensor and a conventional perception system.

Acknowledgments

The research described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

References

Bräunl, T. (2004), "EyeBot — online documentation," The University of Western Australia, http://www.ee.uwa.edu.au/~braunl/eyebot/.

Clark S. M., Durrant-Whyte, H. F. (1997), "The design of a high performance MMW radar system for autonomous land vehicle navigation," *Proc. International Conference on Field and Service Robotics*, pp. 292-299, Canberra ACT, Australia.

Dierauer, P. (1998), "Smart sensors offer increased functionality," *InTech*, pp. 60-63.

Everett, H. R. (1995), *Sensors for Mobile Robots: Theory and applications*, A K Peters, Wellesley, MA.

Frank, R. (2000), *Understanding Smart Sensors*, Artech House, Boston, MA.

Giachino, J.M. (1986), "Smart Sensors," *Sensors and Actuators*, vol. 10, pp. 239-248.

Goldberg, S. B., Maimone, M. W. and Matthies, L. (2002), “Stereo vision and rover navigation software for planetary exploration,” *Proc. IEEE Aerospace Conference*, Big Sky, MT.

Hebert, M. (2000), "Active and passive range sensing for robotics," *Proc. IEEE International Conference on Robotics & Automation*, San Francisco, CA, pp. 102-110.

Howard, A. and Seraji, H. (2001), “Vision-based terrain characterization and traversability assessment,” *Journal of Robotic Systems*, vol. 18, no. 10, pp. 577-587.

Howard, A., Tunstel E., Edwards, D. and Carlson, A. (2001), "Enhancing fuzzy robot navigation systems by mimicking human visual perception of natural terrain traversability", *Joint 9th IFSA World Congress and 20th NAFIPS International Conference*, Vancouver, B.C., Canada, July 2001, pp. 7-12.

Howard, A. and Tunstel, E. (2002). “Development of cognitive sensors,” *NASA Tech Briefs*, 26(4), p. 22.

Horn, B. (1986), *Robot Vision*, MIT Press, Cambridge, MA.

Maimone, M.W., Biesiadecki, J. and Morrison, J. (2001), “The Athena SDM rover: a testbed for Mars rover mobility,” *6th Intl. Symp. on AI, Robotics & Automation in Space*, Montreal, Paper# AM026.

Matthies, L., Balch, T. and Wilcox, B. (1997), "Fast optical hazard detection for planetary rovers using multiple spot laser triangulation," *IEEE International Conference on Robotics & Automation*, Albuquerque, New Mexico, pp. 859-866.

Moore, H. J. et al. (1999), “Soil-like deposits observed by Sojourner, the Pathfinder rover,” *Journal of Geophysical Research*, vol. 104, no. E4, pp. 8729-8746.

Noor, A.K., Doyle, R.J., and Venneri, S.L. (2000), “Autonomous, biologically inspired systems for future space missions,” *Advances in Engineering Software*, vol. 31, pp. 473-480.

Passino, K.M. and Yurkovich, S. (1998), *Fuzzy Control*, Addison Wesley Longman, Menlo Park, CA.

Speissbach, A. J. (1988), “Hazard avoidance for a Mars rover,” *Proc. SPIE Symp. on Mobile Robots III*, vol. 1007, pp. 77-84.

Travis, B. (1995), “Smart Sensors,” *EDN Magazine*, Cahners Publishing.

Tunstel, E. and Howard, A. (2002), “Sensing and perception challenges in planetary surface robotics”, *First IEEE International Conference on Sensors*, Orlando, FL, pp. 1696-1701.

Tunstel, E. and Howard, A. (2003), “Approximate reasoning for safety and survivability of planetary rovers,” *Fuzzy Sets and Systems*, vol. 134, no. 1, 2003, pp. 27-46.

Wilcox, B. (1996), “Nanorovers for planetary exploration,” *AIAA Robotics Technology Forum*, Madison, WI, pp. 11-1-11-6.

Fuzzy Dispatching of Automated Guided Vehicles

***Ran Shneor, +Sigal Berman, *Yael Edan**

*Dept. of Industrial Engineering and Management, Ben-Gurion University of the Negev, POB 653, Beer-Sheva 84105, Israel

+ Dept. of Computer Sciences and Applied Mathematics, Weizmann Institute of Science, Rehovot 76100, Israel

Abstract. Automated Guided Vehicles (AGV) are commonly used to transfer parts in flexible manufacturing systems (FMS). AGV dispatching deals with assigning vehicles to move parts based on the relationship between AGV and parts availability. The dispatching method influences the performance of the FMS and can lead to cost saving and improvement. The production environment is complex and the dispatching decision is sometimes vague and obscure. To deal with such complicated environments a fuzzy logic dispatching model has been developed.

The fuzzy dispatching model was compared to conventional dispatching rules through simulations of different systems. Three types of systems were tested: high volume, medium volume and small volume. Each system was examined with different production rates and AGV numbers. Simulation results indicated that fuzzy dispatching is superior to the other tested dispatching rules, especially in high and medium volume systems.

Key words: fuzzy, dispatching, AGV

1 Introduction

Material handling systems are an important manufacturing component and a key factor in cost savings and efficiency increases (Tompkins 1984). Automated guided vehicles (AGV) are commonly used to transfer parts in flexible manufacturing systems (FMS) (Goetz and Egbelu 1990). AGVs have been credited with admirable benefits including routing flexibility, improved safety and productivity, reduced costs, *i.e.*, material damage (Hoff and Sarker 1998). In addition, AGVs have the advantage of providing automatic interface with other systems (King and Wilson 1991).

AGV system management, *i.e.*, assigning AGVs to delivery jobs is an important layer of AGV operation and has a significant effect on the manufacturing environment (Koff 1987). In dynamic AGV systems AGVs are dispatched during run time using dispatching rules. Such dispatching rules reflect the relationship between the AGV as a resource and the set of parts to be transferred. Many parameters may affect the optimal dispatching choice, *e.g.,* the distance of the AGV from the workstation, the number of parts in outgoing queue. In a real world situation the optimal choice may be vague. Moreover, a different dispatching model may be suitable for different manufacturing settings, *e.g.,* dispatching based on travel distance depends on factory layout. In this paper we advocate a fuzzy dispatching model and conduct a thorough analysis comparing its performance to other models in several diverse manufacturing settings.

A complete methodology for decentralized control of free navigating AGVs was developed and implemented in the computer integrated manufacturing (CIM) laboratory in the Ben-Gurion University of the Negev (BGU) (Berman and Edan 2002). We believe that fuzzy dispatching is most suitable for such a dynamic system.

2 AGV Dispatching

AGV dispatching rules are classified into two categories: work-center initiated rules and vehicle-initiated rules (Egbelu and Tanchoco 1984). Work-center initiated rules are issued by the work-center for selecting between idle AGVs, *e.g.*, the nearest vehicle (NV), the vehicle with the longest idle time (LIV), the least utilized vehicle (LUV). Vehicle-initiated rules are issued by the AGV for selecting what job to perform *e.g.*, shortest travel time/distance (STT/D), maximum outgoing queue size (MOQS), minimum remaining outgoing queue space (MROQS), modified first come first serve (MFCFS). According to the MFCFS rule when a workstation requests an AGV and it cannot be immediately satisfied, the time the request was generated is saved. When an AGV becomes available it is assigned to the workstation with the earliest saved request time.

In recent years many dispatching rules and algorithms were developed based on different parameters of the production environment (Bozer and Yen 1996, Lee 1996, Liu and Duh 1992, Sabuncuoglu and Hommertzheim 1992). Simulations of AGV dispatching rules show that in busy production settings vehicle-initiated rules are dominant since the AGVs hardly remain idle. Work-center initiated rules have little effect in such settings (Egbelu and Tanchoco 1984).

Fuzzy rules were used by several researchers for scheduling (Naumann and Gu 1997, Garbot and Geneste 1994, Custodio *et al* 1994, Tan *et al* 2000). Among the parameters used were processing time, slack time and profitability (Garbot and Geneste 1994). Custodio *et al* (1994) developed a multi layered fuzzy decision system for production planning and scheduling. Tan *et al* (2000) developed a methodology for fuzzy dispatching of AGV's in FMS environment. They used a genetic algorithm for optimal weight selection for the fuzzy rules.

There is little research on fuzzy dispatching and it does not address transporting aspects, especially related to AGV. Naumann and Gu (1997) proposed a real-time part dispatching system within manufacturing cells using fuzzy logic. They used fuzzy variables such as inventory level, time remaining and buffer level in order to eliminate tardiness and to control and minimize buffer levels. Several case studies were conducted providing a more all encompassing approach to part dispatching. Yet there is no comparison between the performance of the material handling system in the tested scenarios and their case studies including part transfer and machine loading times in the total processing time (Naumann and Gu 1997).

3 Fuzzy Dispatching of AGV

The proposed fuzzy dispatching algorithm includes three input variables and a fuzzy inference system. Parameters were chosen to represent three important aspects in production floor management and dispatching: time, money and travel distance. The chosen input variables are: remaining production time (the difference between overall part's production time and current cumulative part's production time), profitability (based on economic and cost accounting data of the part) and travel distance. These inputs were chosen based on empirical test in hardware (Berman and Edan 2002) and fuzzy control surface evaluation using Matlab© logic toolbox.

The fuzzy algorithm enables smooth augmentation of these aspects thus improving decision capabilities in multi objective scenarios such as manufacturing settings. Remaining production time is a parameter incorporated in modern projects management methods, such as theory of constraints (TOC) (Shneor et al. 2002). The shorter the remaining time, the higher the task priority since it is closer to leave the production environment. Profitability has been previously used for AGV dispatching (Garbot and Geneste 1994). Parts with a higher profitability value are given precedence over parts with a lower profitability value. Travel distance is commonly used in dispatching rules especially the shortest travel distance (STD) rule (Egbelu and Tanchoco 1984, Hwang and Kim 1998).

The output of the fuzzy dispatching algorithm is the tasks dispatching priority. Triangular membership functions were selected for the input and output variables based on a preceding simulation analysis (Shneor 2004). Each variable was fuzzified using three levels: low, medium and high (Figure 1). The output was defuzzified using the centroid defuzzification method.

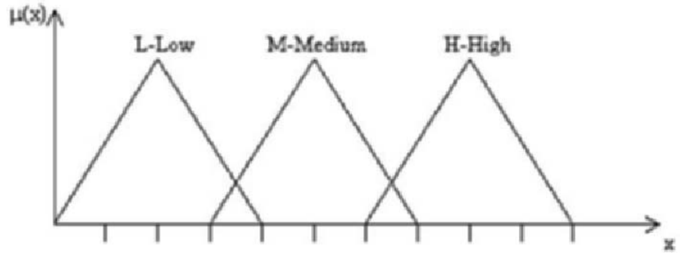


Figure 1. Membership functions

The fuzzy rule base is of full rank and includes 27 rules (Figure 2). The structure of each rule is as follows: If **Time Remaining** is *Tr* and **Profitability** is *Pr* and **Distance** is *D* then Priority is *P*. The fuzzy rule base is knowledge driven and based on analysis of R&D in this area. The rule base and membership function were developed and evaluated using Matlab© (Shneor 2004).

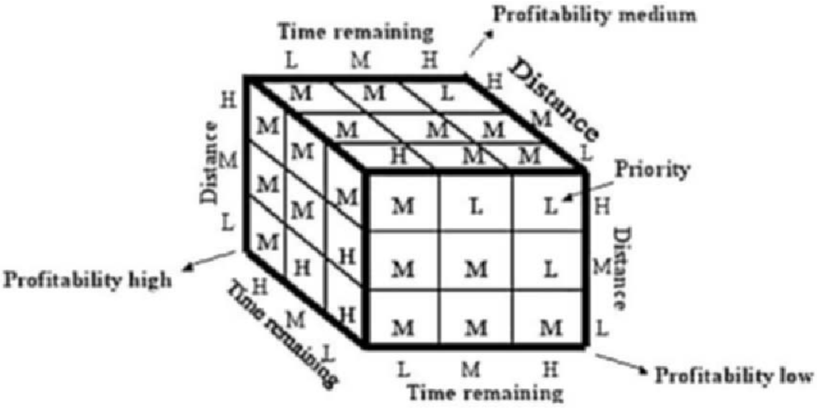


Figure 2. Fuzzy inference rules. Each face of the cube represents a different profitability level (L-low, M-medium, H-high)

4 Simulation Analysis

To evaluate the fuzzy dispatching algorithm and investigate its suitability for different production systems a simulation model was developed. The production system parameters included: system size (number of workstations and products), number of AGVs and parts arrival rate. The operation of each system was evaluated using different dispatching rules. The simulation was developed using Rockwell Arena7© which is a simulation software. Each test case included 30 repetitions of 8 hour production shifts (plus one hour of warm-up time). A total of 432 case studies were conducted.

The simulation was built using the following assumptions:

- Each part has a process plan (including routing and process times) and known sequence of part's production.
- Each production operation must be completed before the next operation can begin.
- Each machine can perform only one operation at a time and each part can be processed by only one machine at a time.
- Each AGV can transfer one part at a time.
- The AGVs use the shortest uni-directional path.
- Each station has an infinite number of buffers for incoming parts and for outgoing parts.
- Each station has a robot for loading/unloading parts from/to AGV.

Three different systems were simulated: high volume consisting of eight stations and ten different part types, medium volume layout containing five stations and five parts and small volume layout including three machines and three parts. Each part is routed through different stations (Table 1). The medium volume system parameters are based on the manufacturing apparatus at BGU (Berman and Edan 2002). Similar systems were tested in previous research (Lee 1996, Egbelu 1987, Garbot and Geneste 1994, Leung and Khtor 1987, Hwang and Kim 1998, Yim and Linn 1993, Custodio et al. 1994, Naumann and Gu 1997, Egbelu and Tanchoco 1984), An overall comparison of all three types of systems described was not considered.

Table 1. Parts routing in the high volume layout (stations 1 and 9 are Automatic Storage Retrieval Systems-ASRS)

	station number								
part 1	1	2	3	4	5	6	7	8	9
part 2	1	8	2	7	3	6	4	5	9
part 3	1	6	2	8	2	7	5	9	
part 4	1	4	7	6	8	2	9		
part 5	1	8	6	4	2	9			
part 6	1	3	5	7	9				
part 7	1	8	6	9					
part 8	1	7	5	3	6	9			
Part 9	1	6	5	4	3	2	9		
Part 10	1	5	8	2	7	4	6	9	

Part arrival rate was exponentially distributed. Two mean parts arrival rates were used (every 0.5 minute, every 1 minute). The number of AGVs needed for each system (Table 2) was determined using the mean of four conventional analytic methods (Egbelu 1987).

Table 2. Results of AGV analytic calculation. Mean numbers for four calculation methods are given.

System	No. of AGV [Egbelu 1987]
Small volume	2-3
Medium volume	4-5
High volume	9-10

Four dispatching policies were compared: Modified First Come First Serve (MFCFS), Maximum Outgoing Queue Size (MOQS), Minimum Remaining Outgoing Queue Space (MROQS) and, Fuzzy Logic Dispatching (FLD). MFCFS, MOQS and MROQS are commonly used dispatching rules and yield reasonable results compared to other dispatching rules in different circumstances (Egbelu and Tanchoco 1984, Bozer and Yen 1996, Liu and Duh 1992, Hwang and Kim 1998, Sinriech and Kotlarski 2002).

The performance measures include the number of parts produced, parts production rate, average order waiting time, average orders queue length, and the number of AGV empty trips (a trip to a designated station without carrying a part). These measures apart from the number of empty AGV trips are commonly used in studies comparing dispatching rules (Kim 1999, Hwang and Kim 1998, Lee 1996, Yim and Linn 1993, Egbelu and Tanchoco 1984). The number of AGV empty trips was used to provide an indication of the performance of the material handling system and enables to analyze the influence of the dispatching rule on the AGV utilization.

5 Results

The results of the simulations conducted are summarized in table 3. In the high volume system the fuzzy dispatching algorithm performed better than the other dispatching rules and led to a higher number of finished parts. The MFCFS rule was secondary and the MROQS led to the lowest results (Figure 3). The average waiting time in orders queue and average orders queue length were lower using fuzzy dispatching, yet as the number of AGV increased the differences between the effects of the dispatching rules decreased. Table 4 shows the number of AGV empty trips in this layout. The number of AGV empty trips were lower using fuzzy dispatching than when using the other rules.

Table 3. Summary results of simulations (FP=Finished Parts, PR=Production Rate, WT=Waiting Time, QL=Queue Length, ET=Empty Trips)

Volume	Type	FP	PR	WT	QL	ET
High (ten AGVs)	MFCFS	652	39	242	39	4062
	MOQS	563	42	396	60	3540
	MROQS	534	46	694	99	3493
	FL	655	38	121	19	571
Medium (four AGVs)	MFCFS	559	45	352	35	2440
	MOQS	473	48	471	41	2055
	MROQS	198	59	900	79	912
	FL	561	42	165	16	356
Small (two AGVs)	MFCFS	587	42	364	24	1664
	MOQS	600	41	421	27	1629
	MROQS	600	41	573	37	1631
	FL	612	41	184	11	235

Table 4. Empty trips of AGV (high volume system, parts arrival: 0.5 min)

No. AGV	**1**		**2**		**3**	
	Average	**+/-**	**Average**	**+/-**	**Average**	**+/-**
MFCFS	921.80	2.51	1725.17	4.47	2190.73	57.56
MOQS	929.73	1.51	1665.50	3.90	2160.03	29.79
MROQS	931.30	2.80	1657.73	5.09	2093.07	25.90
FL	106.97	2.89	215.93	5.04	564.77	83.63

No. AGV	**4**		**5**	
	Average	**+/-**	**Average**	**+/-**
MFCFS	1482.30	101.06	1130.60	83.27
MOQS	1727.23	100.95	1106.90	72.64
MROQS	1703.77	116.28	1160.07	83.70
FL	1022.50	37.18	847.63	46.61

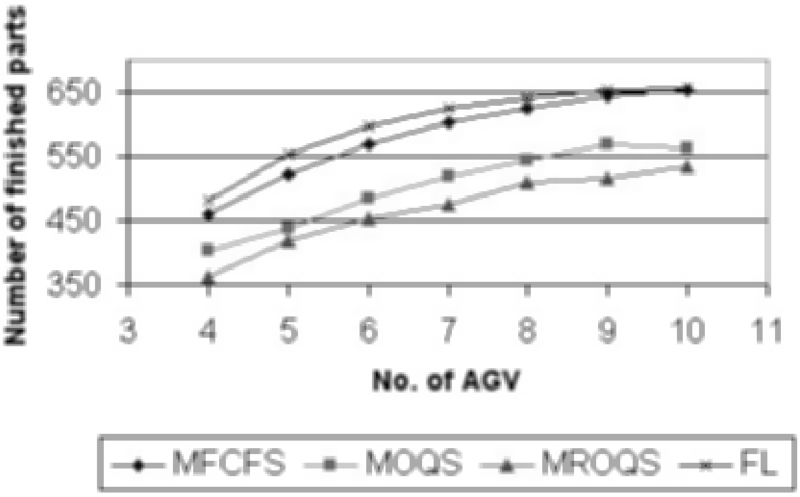


Figure 3. Number of parts produced (high volume system, parts arrival: 0.5 min.)

In the medium volume system the number of finished parts produced and finished parts production rate were better using fuzzy dispatching. The average time in orders queue and the average orders queue length were lower using fuzzy dispatching (Figures 4,5). As the number of AGVs in the medium volume layout increased the order queue length and waiting time decreased. Among the rest of the dispatching rules the MFCFS rule was secondary to the fuzzy dispatching and the MROQS rule yielded poor results. The number of AGV empty trips in this layout was lower with fuzzy dispatching (Figure 6).

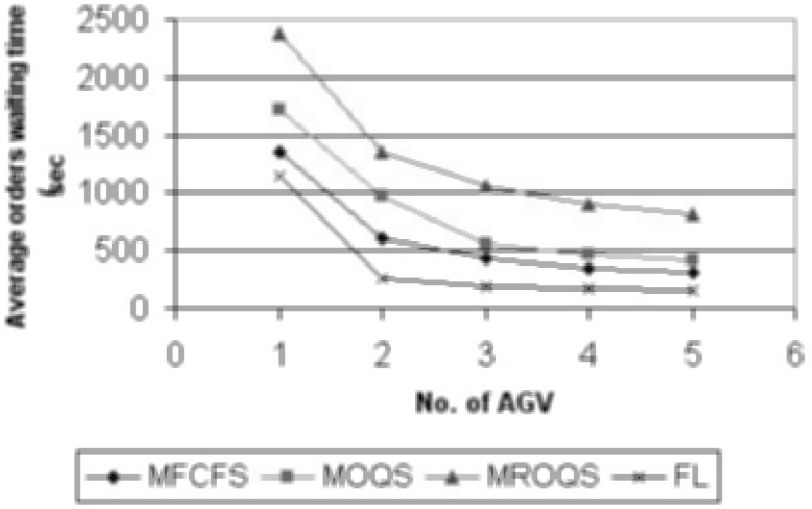


Figure 4. Average waiting time in orders queue (medium volume system, parts arrival: 0.5 min.)

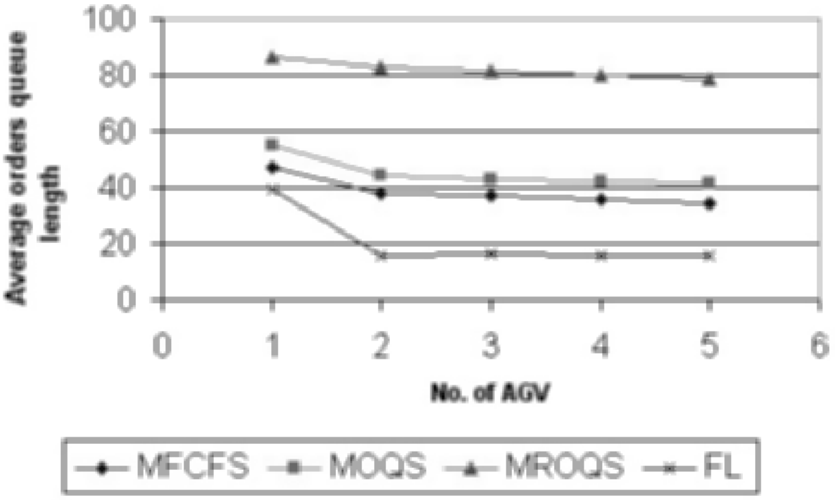


Figure 5. Average orders queue length (medium volume system, parts arrival: 0.5 min.)

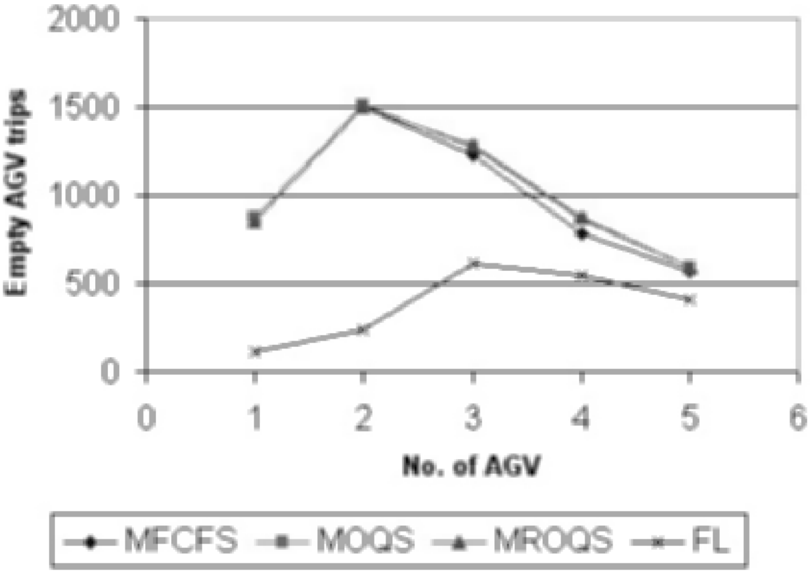


Figure 6: Empty AGV trips (medium volume system, parts arrival: 1 min.)

In the small volume system with low parts arrival rate (every one minute) there were no differences in all measures used between the four dispatching methods. When parts arrival rate was higher (every 0.5 min) there were differences in the performance measures for systems with more than one AGV. In this case fuzzy dispatching outperformed the other dispatching methods for all performance measures.

6 Conclusion

A fuzzy dispatching algorithm was developed and examined in various scenarios using simulation. Analyzing the results shows that fuzzy dispatching performed better than MFCFS, MOQS, MROQS rules. The fuzzy model is superior in high volume, complex systems with several vehicles and high part arrival rate. While the MFCFS, MOQS, MROQS each refer to a single production aspect (queue length, request time) the fuzzy model considers several aspects (time remaining, profitability and distance). Therefore, fuzzy dispatching may be advantageous in complex real world manufacturing scenarios. Another result emerging from the test cases simulations is superiority of the MFCFS and MOQS rule and the inferiority of the MROQS rule.

Our future research endeavors include validating our results in hardware using the BGU CIM system and comparing system operations under additional rules such as shortest travel time or distance. Additionally we intend to test different aspects of the experimental layout such as evaluating the methodology for systems of free navigating AGVs, and for manufacturing system with different production policies *e.g.* push vs. pull systems.

References

Berman, S., Edan, Y., (2002), “Decentralized autonomous AGV system for material handling”, *International Journal of Production Research*, Vol. 40, No. 15, pp. 3995-4006.

Bozer, Y., A., and Yen, C., K., (1996), “Intelligent dispatching rules for trip-based material handling systems”, *Journal of Manufacturing Systems*, Vol. 15, No. 4, pp. 226-36.

Custodio, L., M., M., and Sentieiro, J., J., S., and Bispo, C., F., G., (1994), “Production planning and scheduling using a fuzzy decision system”, *IEEE Transactions on Robotics and Automation*, Vol. 10, No. 2, pp. 160-167.

Egbelu, P., J., (1987), “The use of non-simulation approaches in estimating vehicle requirements in an automated guided vehicle based transport system”, *Material Flow*, vol. 4, pp. 17-32.

Egbelu, P., J., and Tanchoco, J., A., (1984), “Characterization of automatic guided vehicles dispatching rules”, *International Journal of Production Research*, Vol. 22, No. 3.

Garbot, B., and Geneste, L., (1994), “Dispatching rules in scheduling: a fuzzy approach”, *International Journal of Production Research*, Vol. 32, No. 4, pp. 903-915.

Goetz, Jr. W., G., and Egbelu, P., J., (1990), “Guide path design and location of load pick-up/drop-off points for an automated guided vehicle system”, *International Journal of Production Research*, Vol. 28, No. 5, pp. 927-941.

Hoff, E., B., and Sarker, B., R., (1998), “An overview of path design and dispatching methods for automated guided vehicles”, *Integrated Manufacturing Systems*, Vol. 9, No. 5, pp. 296-307.

Hwang H., and Kim, S., H., (1998), “Development of dispatching rules for automated guided vehicle systems”, *Journal of Manufacturing Systems*, Vol. 17, No. 2, pp. 137-143.

Kim, C., W., Tanchoco, J., M., A., and Koo, R., H., (1999), “AGV dispatching base on workload balancing”, *International Journal of Production Research*, Vol. 37, No. 17, pp. 4053-4066.

King, R., E., and Wilson, C., (1991), “A review of automated guided vehicles systems design and scheduling”, *Production Planning and Control*, Vol. 2, No. 1, pp. 44-51.

Koff, G., A., (1987), “Automated guided vehicle systems: applications, controls and planning”, *Material Flow*, vol. 4, pp. 3-16.

Lee, J., (1996), “Composite dispatching rules for multiple-vehicle AGV systems”, *Simulation*, Vol. 66, No. 2, pp. 121-130.

Leung L., C., and Khator, S., K., and Kimbler, D., L., (1987), "Assignment of AGVS with different vehicle types", *Material Flow*, vol. 4, pp. 65-72.

Liu, C., M., and Duh, S., H., (1992), "Study of AGVS design and dispatching rules by analytical and simulation methods", *International Journal of Computer Integrated Manufacturing*, Vol. 5, No. 4 and 5, pp. 290-9.

Naumann, A., and Gu, P., (1997), "Real-time part dispatching within manufacturing cells using fuzzy logic", *Production planning & control*, Vol. 8, No. 7, pp. 662-669.

Sabuncuoglu, I., and Hommertzheim, D., L., (1992), "Dynamic dispatching algorithm for scheduling machines and automated guided vehicles in a flexible manufacturing system", *International Journal of Production Research*, Vol. 30, No. 5, pp. 1059-79.

Sinriech, D., and Kotlarski J., (2002), "A dynamic scheduling algorithm for a multiple-load multiple-carrier system", *International Journal of Production Research*, Vol. 40, No. 5, pp. 1065-1080.

Shneor, R., Avraham, R., Amar, S., Shitrit, Y., (2002), "Implementing project management according to TOC in aerial maintenance unit", *12th IE&M National Conference*, Tel-Aviv, Israel.

Shneor, R., (2004), "Fuzzy dispatching of AGV", *M.Sc thesis*, Ben-gurion university, Beer-sheva, Israel.

Tan, K., K., and Tan, K., C., and Tang, K., Z., (2002), "Evolutionary tuning of a fuzzy dispatching system for automated guided vehicles", *Systems, Man and Cybernetics, Part B, IEEE Transactions on*, Vol. 30, No. 4, pp. 632-636.

Tompkins, White, Bozer, Frazelle, Tanchoco, Trevino, (1984), *Facilities Planning*, 2nd ed., John Wiley & Sons, New York.

Yim, D., and Linn, R., J., (1993), "Push and pull rules for dispatching automated guided vehicles in a flexible manufacturing system", *International Journal of Production Research*, Vol. 31, No. 1, pp. 43-57.

Part XVIII

Soft Computing and Hybrid Intelligent Systems in Product Design and Development

Application of Evolutionary Algorithms to the Design of Barrier Screws for Single Screw Extruders

A. Gaspar-Cunha, L. Gonçalves, J.A. Covas

IPC – Institute for Polymers and Composites,
Dept. of Polymer Engineering, University of Minho,
Guimarães, Portugal
{agc, lfgoncalves, jcovas}@dep.uminho.pt

Abstract. An optimization approach based on Multi-Objective Evolutionary Algorithms (MOEA) - the Reduced Pareto Set Genetic Algorithm with Elitism (RPSGAe) - is used to design barrier screws for single screw extruders. A numerical modeling routine able to describe the thermo-mechanical experience suffered by the polymer inside the extruder is developed. A specific screw design methodology is proposed and used to optimize conventional and barrier screws. In this methodology, the fittest screw geometry is chosen after performing a sensitivity study of a set of "best" screws to limited changes in operating conditions.

Keywords: multi-objective optimization, evolutionary algorithms, polymer extrusion, barrier screws.

1.1 Introduction

Polymer extrusion is one of the most important manufacturing technologies of plastics parts. Products such as films, sheets, pipes, profiles, electrical wires and filaments are usually made using extruders. An extruder consists, basically, of an Archimedes-type screw rotating at constant frequency inside a heated barrel (see Figure 1). The solid polymer, usually in pellet form, is deposited in the hopper and flows by gravity towards the lateral barrel opening. Then, by action of the screw rotation, the polymer is dragged forward, melts progressively and is pressurized, in order to flow though the die that yields the final shape of the product.

Intense experimental work carried out during the last four decades revealed that the process can be subdivided in six adjacent individual functional zones, as illustrated in Figure 1-A (Tadmor and Klein 1970, Agassant et al. 1996, Rauwendaal 1996): granular gravity solids conveying in the hopper, friction drag solids conveying in the screw, conveying of solids (partially) surrounded by a melt film, melting according to a specific melting mechanism, melt conveying in the last turns of the screw and pressure flow through the die. Therefore, modeling of the extrusion process from hopper to die involves the resolution of the governing equations of each step of the extrusion process, for a specific combination of polymer properties, screw geometry and operating conditions, coupled to the relevant boundary conditions (Gaspar-Cunha 2000). One can then predict the extruder response in terms of major parameters such as mass output, power consumption, melt temperature at die exit, degree of mixing and viscous dissipation. This is known as solving the direct problem.

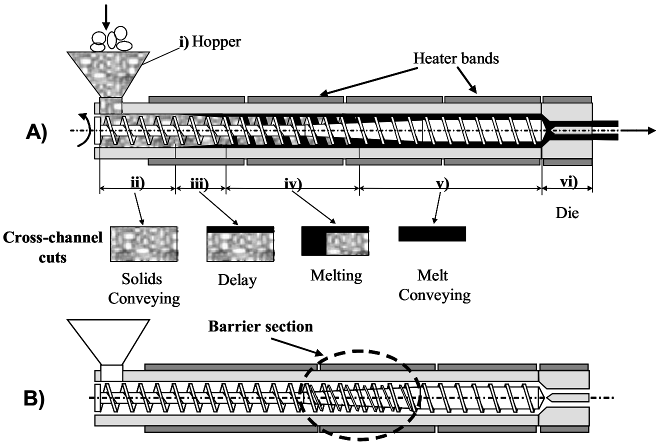


Fig. 1. Single screw extrusion. A) Conventional layout and corresponding functional zones; B) Barrier screw.

Process modeling and practical experience have evidenced some important limitations of conventional screws, namely their melting efficiency. This led to the introduction of barrier screws (Figure 1-B), which contain a section where an additional flight is introduced. The aim is to increase the contact area between the solids and the metallic walls of the extruder and to separate the melt from the solids in order to improve process stability.

Screw design and/or the definition of the operating conditions involve solving the inverse problem, *i.e.*, solving the governing equations in order to the geometrical parameters or the operating conditions, using the prescribed performance as boundary conditions. Unfortunately, this is not only too complex but also ill posed, as there is no explicit unique relationship between cause and effect. Thus, in practice, a trial-and-error procedure is adopted instead, the user defining the screw geometry and/or operating conditions that are fed to the modeling routine, which provides the corresponding process response. It is then the user who assesses the related performance and who eventually defines a new tentative screw and/or operating conditions. Alternatively, the inverse problem can be considered as an optimization problem where the goal is to maximize an objective function that takes into account the global process performance (Gaspar-Cunha 2000, Covas et al. 1999, Gaspar-Cunha et al. 2001). Consequently, the main objectives of this work are to develop a general optimization methodology and to implement a (automatic) design strategy of barrier extrusion screws.

1.2 Process Modeling

The modeling routine considers the sequence of individual zones identified in Figure 1A, the calculations being performed at small screw and die increments (Gaspar-Cunha 2000). Solids conveying assumes the displacement of a non-isothermal solid plug with heat dissipation at all surfaces (Gaspar-Cunha 2000). A melt film near to the inner barrel wall (with heat convection on the radial and longitudinal directions) will eventually appear, followed by a solid bed surrounded by melt films (Gaspar-Cunha 2000). Melting is supposed to follow the 5-zone model proposed by Lindt *et al.* (1985). Finally, a 2D non-isothermal non-Newtonian flow develops in the melt conveying zone and die (Gaspar-Cunha 2000). The complexity of computing the effect of the existence of the barrier is mainly associated with its axial location and length, i.e., whether melting and melt conveying are initiated before, during, or after the barrier. The resulting computational procedures have been discussed recently (Gaspar-Cunha et al. 2002).

Various types of conventional and barrier screws can be selected (see Figure 2). Usually, the modeling assumes that the polymer flows on a rectangular channel obtained by the unfolding of the screw channel, as represented in Figure 2. The conventional screw has three different sections. For sections 1 and 3 the channel depth is constant and equal, respectively, to H_1 and H_3 (H_1•H_3), while for section 2 the channel depth decreases from H_1 to H_3. In this case the parameters to optimize are the length of sections 1 and 2 (L_1 and L_2), the channel depth of sections 1 and 3

(H_1 and H_3), the screw pitch (P) and the flight width (e). The geometry of the Maillefer barrier screw is identical to that of the conventional screw, except on the existence of a barrier separating the solid from the melt. Thus, the Maillefer screw has two more parameters to optimize, the width of the barrier (W_f) and the barrier clearance (H_f), i.e., the distance between the barrier and the barrel. The geometry of the remaining three barrier screws is more complicated and they will not be considered in this study. The Barr barrier screw is constituted by 5 different sections. In this case, sections 1 and 5 are similar to sections 1 and 3 of the conventional screw, sections 2 and 4 are similar to section 2 of Maillefer screw and section 3 is characterized by solid and melt channels with constant width, but while the depth of the solids channel decreases the depth of the melt channel increases. The Dray and Lawrence screw use the same concept to that of the Barr screw, i.e., the increase on the melt channel area is obtained at expenses of an increase on the depth of the melt channel. Finally, the increase of the melt channel area of the Kim screw is due to an increase in both the width and the depth of the channel.

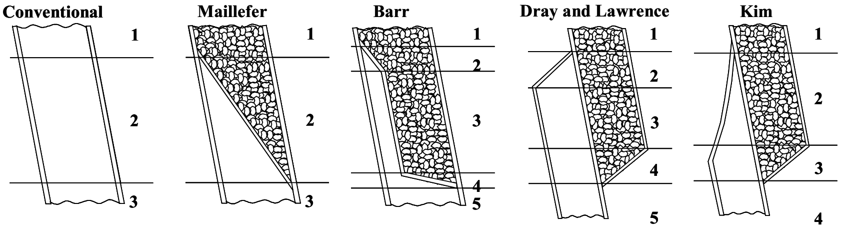


Fig. 2. Conventional and barrier screw-types.

1.3 Design Approach

1.3.1 Multi-Objective Optimization Algorithm

Real optimization problems are generally multi-objective, *i.e.*, they require the simultaneous optimization of various (often conflicting) criteria (Gaspar-Cunha et al. 2002). The solution must then result from a compromise between the different criteria. Traditionally, these are aggregated in a global objective function, such as the weighted sum (*i.e.*, the sum of the individual objective functions affected by a weight quantifying their relative importance in the process). This approach has often been criticized, not only because the magnitude of every criteria must be fixed before the search is carried out, but also because in the case of mixed (simultaneous maximization and minimization) and non-convex problems, different weights can yield the same optimum (Gaspar-Cunha et al. 2002). Instead, multi-objective optimization using EAs (MOEA) may be used, where the objective is to obtain the set of points expliciting the trade-off between the criteria represented, designated as Pareto frontier. In this case there is no need to define *a priori* the individual weights and a single run provides all the possible compromises between the criteria (Gaspar-Cunha et al. 2002).

A MOEA must accomplish two basic tasks: i) guiding the population in the direction of the Pareto frontier and ii) maintaining the solutions dispersed along the entire Pareto. The Reduced Pareto Set Genetic Algorithm with elitism (RPSGAe) is adopted here (Gaspar-Cunha et al. 2003). It is based on a clustering technique used to reduce the number of solutions on the efficient frontier. Fitness is determined through a ranking function, the individuals being sorted using that technique. Initially, an internal and an empty external population are created. At each generation, the clustering algorithm allows the selection of a fixed number of best individuals to be copied to the external population. When the latter becomes full, the RPSGAe is applied to sort its individuals. Finally, the best individuals of the external population are incorporated in the internal one by replacing the lowest fitness individuals (Gaspar-Cunha et al. 2003). Detailed information about this algorithm can be found elsewhere (Gaspar-Cunha 2000, Gaspar-Cunha et al. 2003):

```
1- Random initial population (internal);
2- Empty external population;
3- while (not Stop-Condition) do
  a- Evaluate internal population;
  b- Calculate the Fitness using the RPSGAe;
  c- Copy the best individuals to the external pop.;
  d- if (the external population becomes full)
    - Apply the RPSGAe to this population;
    - Copy the best individuals to the internal
pop.;
  end if
  e- Select the individuals for reproduction;
  f- Crossover;
  g- Mutation;
end while
```

1.3.2 Methodology for Screw Design

The application of the above optimization methodology, in designing screws for polymer extrusion, must take into account the different number of parameters to optimize necessary by the diverse types of screws (CS, MS, BS, DLS and KS). The difficulty, in using an Evolutionary Algorithm to optimize this problem, appears due to the variable length of the chromosome. Thus, the usual binary representation of the parameters to optimize cannot be used. As an alternative, a structured chromosome representation is adopted (Dasgupta and McGregor 1992). This type of structure is characterized by the existence of redundancy (*i.e.*, various part of the chromosome representing the same variables) and a hierarchy in the genotype representation. Figure 3 illustrates a possible structured representation for the case of design of screw extrusion. This representation is constituted by 3 hierarchy levels. The first level (*i.e.*, the higher level) identifies the type of screw, a_1 represents a conventional screw (CS) and a_2 represents a barrier screw. In this case the redundancy conflict is eliminated by the value of a_1 and a_2, which operate as regulatory genes. For example, if "a_1a_2" is represented by "10", then a conventional screw must be considered, together with the respective geometrical parameters (L_1, L_2, H_1, H_3, P and e). On the contrary, if "a_1a_2" is represented by "01", then a barrier screw is to be considered. In this case 4 different possibilities arise, *i.e.*, screws MS, BS, DLS or KS are to be considered, together with the specific geometrical parameters. Therefore, the behavior of this representation is governed by the high level genes, which activate or deactivate the lower level ones.

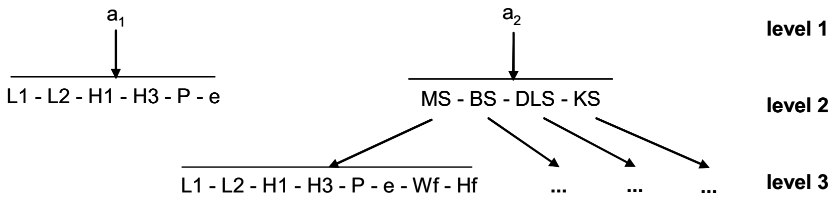

a1
a2
level 1
L1 - L2 - H1 - H3 - P - e
MS - BS - DLS - KS
level 2
L1 - L2 - H1 - H3 - P - e - Wf - Hf
...
...
...
level 3

Fig. 3. Global chromosome structure.

An important aspect to take into account is the linkage between the EA and the modeling routine, mainly when this chromosome structure must be transformed in the geometrical parameters to be optimized. The modeling routine must be able to deal with various types of screws simultaneously, often with very different geometry. In this study the problem was simplified by maintaining some of the geometrical parameters constant, such as the internal barrel diameter and the total screw length. In the industrial practice this is not a real problem, since the manufacturer have an extruder with a barrel of fixed dimensions and what he wants is to define the geometry of the screw.

Finally, the actual optimization methodology for screw design must consider the required flexibility of these devices in use, *i.e.*, they must perform satisfactorily within a range of operating conditions and material properties. Therefore, as shown in Figure 4, after performing an optimization run, a set of the most performing screws is subjected to a sensitivity analysis in terms of limited changes in operating conditions, polymer rheological behavior and criteria. The most stable screw will be chosen.

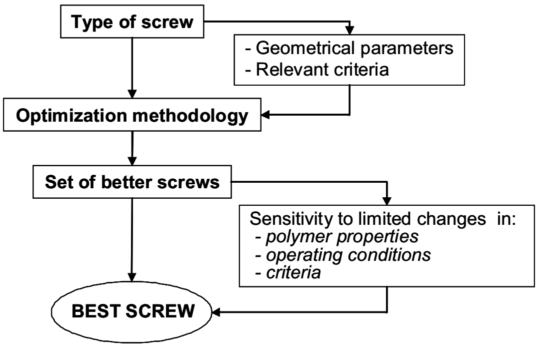


Fig. 4. Screw design methodology.

1.4 Optimization Example

In order to illustrate the working of the design methodology proposed, an example where the aim consists in defining the feed (L_1) and barrier (L_2) screw lengths, and the screw depth of the feed (H_1) and metering (H_3) sections, will be used. For that purpose four runs are carried out using both, conventional and Maillefer barrier screws (see Table 1). The aim is to maximize output and minimize the mechanical power consumption for runs 1 to 3 and to maximize output and degree mixing and minimize melt temperature, melting length and mechanical power consumption for run 4. The operating conditions are kept constant (screw speed of 60 rpm and uniform barrel temperature of 170ºC). Runs 1 and 2 will be used to optimize individually a conventional and a Maillefer screws, respectively. In runs 3 and 4 the structure defined in Figure 5 will be used to optimize simultaneously conventional and Maillefer barrier screws.

Table 1. Type of screw and criteria used in the optimization runs.

	Type of Screw	Criteria
Run 1	CS	Q, Power
Run 2	MF	Q, Power
Run 3	CS + MS	Q, Power
Run 4	CS + MS	Q, WATS, Length, Texit, Power

Figure 6 shows the Pareto-frontiers obtained for runs 1, 2 and 3. As expected, Maillefer barrier screw (MS) is able to reach higher outputs, since it is able to melt a larger quantity of polymer before the end of the screw. This is attained at costs of increasing the mechanical power consumption. In fact, the mass output and the mechanical power consumption are conflicting criteria because the mechanical power consumption is proportional to the output. Thus, for small values of output the mechanical power consumption is smaller but this is better accomplished by the conventional screws (CS). Figure 6 compares, also, the results obtained for run 3, where the chromosome structure of Figure 5 is used. In this case the aim is to optimize conventional and Maillefer screws simultaneously. The Pareto curve obtained shows that conventional screws are better when small output and mechanical power consumption are to be obtained and Maillefer screws are better for maximizing output.

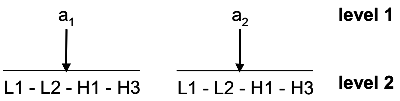


Fig. 5. Chromosome structure for runs 3 and 4.

A great advantage of this analysis consists in the possibility of the industrialist to have a global visualization of all the relevant data. For that purpose is possible to represent the Pareto curves in the parameters to optimize domain. With this type of representation the industrialist is able to define regions of the parameters to optimize space and relate this with the optimization criteria (Figure 6). For example, the differences observed on the criteria domain for the MBS are accomplished by changes on D1 and D2 variables, since the values of L1 and L2 converges for its maximum possible value. On the contrary, the differences for the CS are accomplished by changes in L1 and L2.

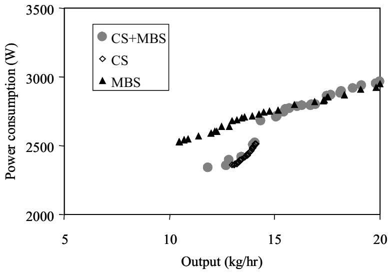


Fig. 6. Optimization results for runs 1 (CS), 2 (MS) and 3 (CS+MBS).

Applying the structure of Figure 5 to the same problem, but now considering five criteria simultaneously (run 4 – Table 1), produces the Pareto frontiers represented in Figure 7, where all non-dominated solutions are plotted in terms of the various criteria against output. Figure 8 presents the Pareto plots in the parameters to optimize domain. The optimal conventional screws (CS) obtained have big H_1 values and small H_3 values. The Pareto frontiers are a cloud of points and not a line, since this is a multidimensional space and, for that reason, points that seem to be dominated in one Pareto plot may be non-dominated in another. Also, it is not easy to find the best solutions. However, if, for example, in a specific application the degree of mixing is defined as the most important criterion, the five screws presented in Table 2 are to be selected. This includes 3 Maillefer and two conventional screws. Taking into account that the screw to be selected must be stable to variations on the operating conditions used, they will be subjected to sensitivity studies to changes in operating conditions.

Figure 9 compares the stability of each screw to small changes in operating conditions. For that purpose, combinations of three screw speeds (55, 60 and 65 rpm) and three barrel temperature profiles (165, 170 and 175 ºC) were used. The average and standard deviation of the global objective function are also represented. The global performance of screws 1, 2 and 5 (i.e., Maillefer screws) is better than the global performance of screws 3 and 4 (i.e., conventional screws). The stability, quantified by the standard deviation, is quite the same for all screws. Thus, the choice must be made between screws 1, 2 and 5.

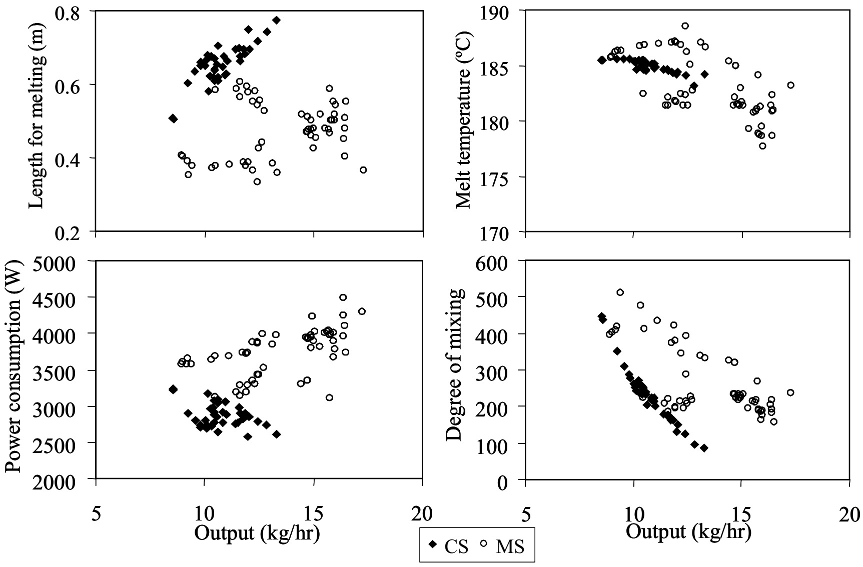


Fig. 7. Results for the example of Figure 6 using the MOEA approach.

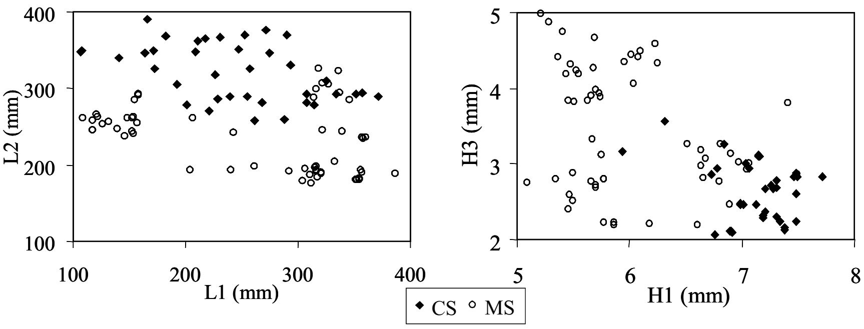


Fig. 8. Pareto frontiers in the parameters to optimize domain for run 4.

1.5 Conclusions

The design methodology proposed here for the design of barrier screws for polymer extrusion, combining optimization with EAs and sensitivity analyses, seems to be potentially useful in industrial practice, as it provides solutions with satisfactory physical meaning. However, and despite its obvious difficulty, experimental validation must be carried out. Also, the method incorporates practical knowledge, such as criteria type and relative relevance, or ranges of variation of the geometrical parameters. These must be defined carefully, as they influence strongly the results obtained.

The current limitations of the design method are mainly related to the modeling capacity (for example, it is not easy to model the process dynamic stability, or to establish a sufficiently accurate correlation between machining costs and geometrical features) and the computational requirements.

Table 2. Type of screw and criteria used in the optimization runs.

	Type	L1 (mm)	L2 (mm)	L3 (mm)	H1 (mm)	H3 (mm)
Screw 1	MS	122	263	501	5.77	2.23
Screw 2	MS	127	254	506	5.46	2.40
Screw 3	CS	108	349	429	7.38	2.12
Screw 4	CS	107	349	431	7.38	2.17
Screw 5	MS	132	257	497	5.47	2.60

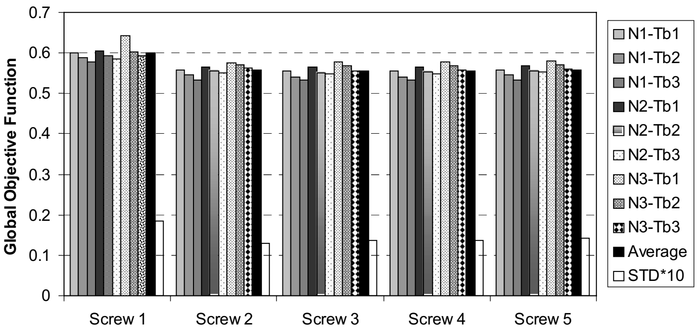


Fig. 9. Sensitivity to small changes in Operating conditions (N1 = 55 rpm, N2 = 60 rpm, N3 = 65 rpm, Tb1 = 165 ºC, Tb2 = 170 ºC, Tb3 = 175 ºC).

Acknowledgments

The Portuguese Foundation for Science and Technology provided partial financial support for this work under the program POCTI/EME/ 48448/2002.

References

Agassant, J.F., Avenas, P., Sergent, J. (1996), La Mise en Forme des Matiéres Plastiques, 3rd edn. Lavoisier, Paris.

Covas, J.A., Gaspar-Cunha, A., Oliveira, P. (1999), An Optimization Approach to Practical Problems in Plasticating Single Screw Extrusion, Polym. Eng. and Sci., vol. 39, pp. 443-456.

Dasgupta, D., McGregor, D.R. (1992), Nonstationary Function Optimization Using Structured Genetic Algorithm, Proceedings of Parallel Problem Solving from Nature (PPSN-2), pp. 145-154.

Deb, K. (2001), Multi-Objective Optimization using Evolutionary Algorithms, Wiley.

Gaspar-Cunha, A. (2000), Modelling and Optimisation of Single Screw Extrusion, PhD Thesis, University of Minho, Guimarães.

Gaspar-Cunha, A., Covas, J.A. (2001), The Design of Extrusion Screws: An Optimisation Approach. Intern. Polym. Process., vol. 16, pp. 229-240.

Gaspar-Cunha, A., Covas, J.A. (2004), - RPSGAe - A Multiobjective Genetic Algorithm with Elitism: Application to Polymer Extrusion, in a Lecture Notes in Economics and Mathematical Systems volume, Springer.

Gonçalves, L.F., Gaspar-Cunha, A., Covas, J.A. (2003), Modelling and Design of Barrier Screws for Single Screw Extruders, PPS- Europe-Africa Meeting, Athens, Greece.

Lindt, J.T., Elbirli, B. (1985), Effect of the Cross-Channel Flow on the Melting Performance of a Single-Screw Extruder, Polym. Eng. Sci., vol. 25, pp. 412-418.

Rauwendaal, C. (1990), Polymer Extrusion, 2nd edn. Hanser Publishers.

Tadmor, Z., Klein, I. (1970), Engineering Principles of Plasticating Extrusion, Van Nostrand Reinhold, New York.

Soft Computing in Engineering Design: A Fuzzy Neural Network for Virtual Product Design

Xuan F Zha*

National Institute of Standards and Technology, USA

Abstract. This paper presents a fuzzy neural network approach to virtual product design. In the paper, a novel soft computing framework is developed for engineering design based on a hybrid intelligent system technique. First, a fuzzy neural network (FNN) model is proposed for supporting modeling, analysis and evaluation, and optimization tasks in the design process, which combines fuzzy logic with neural networks. The developed system provides a unified soft computing design framework with computational intelligence. The system has self-modifying and self-learning functions. Within the system, only one network is needed training for accomplishing the evaluation, rectification/modification and optimization tasks in the design process.

Keywords: soft computing, computational intelligence, intelligent design, design process, design pattern, integration, virtual product design, neural networks, and fuzzy logic

1. Introduction

Design process is an iterative and highly interactive task. The designer has to consider countless constraints with usually opposing goals during the design process. Automatic design, analysis, evaluation, modification and optimization of designed parameters are important issues to be addressed in design process. There are many individual intelligent approaches such as expert system, fuzzy logic, neural network and genetic algorithm that can accomplish these tasks (Dagli 1994, Wang and Takefuji 1993, Zha 2004b). However, due to particular computational properties of individual intelligent system techniques, hybrid solutions in the intelligent systems community are required to solve complex design problems, which may integrate one or more of the above individual intelligent techniques (Jang 1993, Medsker 1995, Goonatilake and Khebbal 1995, Zha 2004a-b).

In recent years, virtual product design is an emerging technology that allows engineers to visualize multi-dimensional properties of new product at its design stage without actually producing a prototype (Bateman 1997, Law 1991, Li et al. 2000). A key issue of virtual product design is to accurately correlate the simulation output with the input conditions and after that to evaluate/predict the machine's behavior and performance under new conditions.

* Email: zha@cme.nist.gov

Conventional evaluating/predicting methods suffer from several deficiencies and limitations, which make them inadequate for virtual product design in today's manufacturing requirements. It is, therefore, desirable to have a new evaluating/predicting method that can overcome these deficiencies and limitations. Recently, there have been many attempts to solve these problems by the applications of neural networks (Azoff 1994, Kang 1991, Refenes 1995, Zha 2003, 2004a). The strengths of neural networks accrue from the fact that they need not *priori* assumptions of models and from their capability to infer complex, nonlinear underlying relationships. From the statisticians' point of view, neural networks are essentially statistical devices for performing inductive inference and are analogous to non-parametric, nonlinear regression models (Zurada 1992). However, existing neural schemes use two or more separate neural networks to accomplish some design tasks respectively, and need to train them separately. This is tedious and costly, and sometimes very difficult. In order to overcome the suffered shortcomings or difficulties above, more research endeavors are necessary to develop more general topologies of neural models, learning algorithms and approximation theories so that those models are applicable in the system modeling and control of complex systems. A new kind of hybrid neural networks is therefore required for design support.

This paper aims to develop an intelligent tool using fuzzy neural network (FNN) model to support virtual product design. The remaining parts of this paper are organized as follows. Section 2 proposes an FNN model. Section 3 outlines the implementation of the FNN design system. Section 4 presents a simple case study using the developed model and system. Section 5 summarizes the paper.

2. Soft Computing Framework for Engineering Design

Product development is determined mainly by the design of a product and its process. In this work, a soft computing framework, as shown in Fig.1, is proposed for design with hybrid intelligent integration of soft computing techniques and knowledge-based expert system as well as computer-aided design (CAD). The principal constituents of soft computing are fuzzy logic, neuro-computing, evolutionary computing and probabilistic computing, with the later subsuming belief networks, chaotic systems and parts of learning theory. Soft computing techniques facilitate the use of fuzzy logic, neuro-computing, evolutionary computing and probabilistic computing in combination, leading to the concept of hybrid intelligent systems. Therefore, a systematic method for the design of product-process requires the concurrent integration of the following processes or procedures: product design (CAD), design for operation, process planning, task assignment and line balancing, equipment selection, production system design/layout, evaluation and simulation, etc.

3. Fuzzy Neural Network Model

In this work, a fuzzy neural network (FNN) model is proposed for supporting modeling, analysis and evaluation, and optimization tasks in the design process in the above soft computing design framework, which combines fuzzy logic with neural networks. Details about the fuzzy neural network model are discussed in this section.

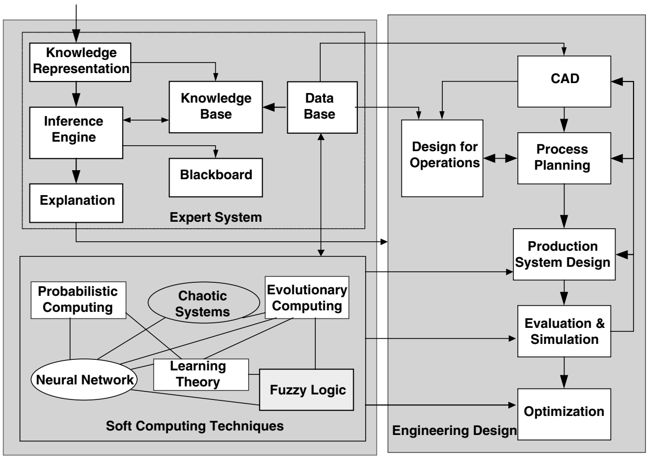


Fig. 1. Soft computing engineering design framework

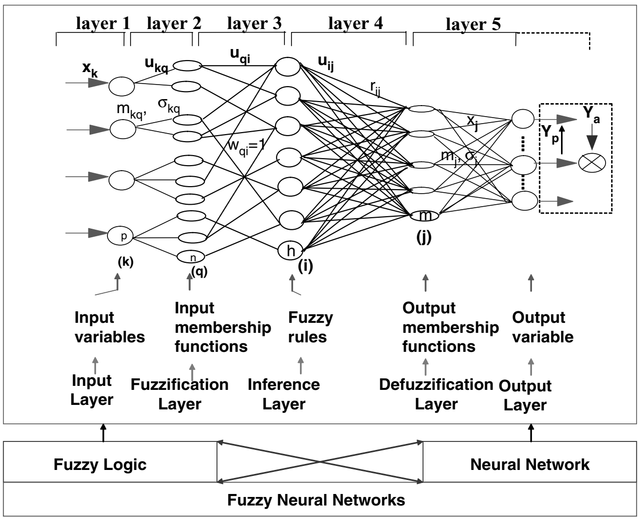


Fig. 2. The architecture of FNN

3.1 The FNN Architecture

Fig.2 shows the architecture of the Fuzzy Neural Network (FNN) which is basically a five-layer fuzzy rule-based neural network (Li et al 2000). In accordance with the common neural network notation, a node in any layer *n* of the network has its input termed *net-in*$^{(n)}$ or simply *net*$^{(n)}$. The node performs a certain operation on the input and generates an output which is a function of its input, i.e., output $f(net^{(n)})$.

Layer 1: The nodes in this layer transmit input values x_k to next layer directly as u_{kq}, where $x_k = u_{kq}$. That is,

$$\mathrm{net}_k^{(1)} = x_k\,;\quad f(\mathrm{net}_k^{(1)}) = \mathrm{net}_k^{(1)} = u_{kq} \tag{1}$$

where $k = 1, 2, \ldots, p$; $q = 1, 2, \ldots, n$

Layer 2: The nodes in layer 2 are the input membership functions. They work as a fuzzifier transforming a numerical input into a fuzzy set. The membership functions are normal distributions with a range between 0 and 1.

$$net_q^{(2)} = -\frac{(u_{kq} - m_{kq})^2}{(\sigma_{kq})^2} \quad \text{and} \quad f(net_q^{(2)}) = e^{net_q^{(2)}} = u_{qi} \tag{2}$$

where $i = 1, 2, \ldots, h$

where, m_{kq} and σ_{kq} are the means and variances of the input membership functions, respectively.

Layer 3: The nodes in this layer perform a fuzzy min-max operation on the node inputs, i.e. a fuzzy AND operation followed by a fuzzy OR operation.

$$net_i^{(3)} = min\{u_{qi} \cdot w_{qi}\} \quad \text{and} \quad f(net_i^{(3)}) = net_i^{(3)} = u_{ij}\text{, where } i = 1, 2, \ldots, h$$

$$u_{cj} = max(net_1^{(3)}, net_2^{(3)}, \ldots, net_h^{(3)}) \tag{3}$$

where, $j = 1, 2, \ldots, m$ and $c \in \{1, 2, \ldots, h\}$. The link weight w_{qi} is unity. The node *c* is termed the winner node of the fuzzy min-max operation.

Layer 4: The links in this layer represent the dampened outputs of the winner node.

$$net_j^{(4)} = u_{cj} \cdot r_{cj} \quad \text{and} \quad f(net_j^{(4)}) = net_j^{(4)} = x_j \qquad (i = \mathrm{c}) \tag{4}$$

$j = 1, 2, \ldots, m$

The dampening coefficients are the rule values r_{ij}. The initial rule values are either random values or assigned directly by an expert. They can also be established outside the network from historical data and then incorporated into the network. The rule values are subsequently fine-tuned during learning.

Layer 5: This layer performs defuzzification of outputs. The defuzzification method used is the centre of gravity method (Kosko 1992), which uses the centroid of the membership function as the representative value. Thus if m_j and σ_j are the means and the variances of the output membership functions respectively, then the defuzzified outputs are given by Eq.(5) as follows:

$$\hat{y}(t) = f(net_{output}^{(5)}) = \frac{net_{output}^{(5)}}{\sum_{j=1}^{m} m_j x_j} \tag{5}$$

where $net_{output}^{(5)} = \sum_{j=1}^{m} □_j m_j x_j$ and $m_j \bullet \sigma_j$ are the link weights, j =1,2 , ...,m.

3.2 Parameter Learning Algorithms

Fuzzy neural networks can be trained using various training/learning methods, such as the back propagation method (Horikawa et al. 1992), the Kohonen's feature maps method (Kohonen 1988), and the conjugate gradient method (Leonard and Kramer 1990). Standard back propagation remains the most widely used supervised learning method for neural networks (Rumelhart et al. 1986). In the following, the first two algorithms, i.e., the back-propagation algorithm and the Kohonen's feature maps method (Xiang et al. 2001, Li et al. 2000) are given.

3.2.1 The Supervised Learning

The objective of the supervised learning is to minimize the error function *ER* as defined in Eq. (6) below by means of a learning algorithm.

$$E = \frac{1}{2}[y(t) - \hat{y}(t)]^2 \tag{6}$$

where, $y(t)$ is the actual output, and $\hat{y}(t)$ the predicted output. In FNN, the learning algorithm used is derived from the back-propagation algorithm (Rumelhart 1986). Thus if η is the assigned learning rate, the rule values r_{ij} are fine-tuned as follows:

$$r_{ij}(t+1) = r_{ij}(t) - \eta \frac{\partial E}{\partial r_{ij}} \tag{7}$$

$$\frac{\partial E}{\partial r_{ij}} = \frac{\partial E}{\partial f(net_j^{(4)})} \cdot \frac{\partial f(net_j^{(4)})}{\partial (net_j^{(4)})} \cdot \frac{\partial (net_j^{(4)})}{\partial r_{ij}} \tag{8}$$

From Eqs. (4) and (5), we have

$$r_{ij}(t+1) = r_{ij}(t) - \eta \frac{\partial E}{\partial r_{ij}} = r_{ij}(t) + \eta u_{ij}[Y_a(t) - Y_p(t)] \frac{(\sum_{j=1}^{m} \sigma_j x_j)\sigma_j m_j - (\sum_{j=1}^{m} \sigma_j m_j x_j)\sigma_j}{(\sum_{j=1}^{m} \sigma_j x_j)^2} \tag{9}$$

where ,

$u_{ij} = min[e^{-\frac{(x_k - m_{kq})^2}{(□\sigma_{kq})^2}}]$, the *net-in* to the node of layer 4

x_k = Values of input in the 1st layer, k = 1,2,...,K

m_{kq} = Mean of input membership function in the 2nd layer

�$_{kq}$ = Variance of input membership function in the 2nd layer

η = Learning rate

r_{ij} = Rule value or damping coefficient

$Y_a(t)$ = Actual output

$Y_p(t)$ = Predicted output

m_j = Mean of output membership function in the 5th layer

σ_j = Variance of output membership function in the 5th layer

x_j = Net-input to the layer-5 node

The learning process is iterated until an acceptable minimum error between actual output Y_a and predicted output Y_p is achieved.

3.2.2 Self-Organized Learning

The Kohonen's Feature Maps algorithm (Kohonen 1988) is used here to find the number of membership functions and their respective means and variances. The algorithm is explained below:

For a given set of data X = ($x_1, x_2, \ldots, x_n$), initial mean values $m_1, m_2, \ldots, m_k$ are assigned arbitrarily, where
min ($x_1, x_2, \ldots, x_n$) < m_i < *max* ($x_1, x_2, \ldots, x_n$).The data are then grouped around the initial means according to:

$$|x_j - m_c| = \min_{i} \{ |x_j - m_i| \} \qquad 1 \le i \le k \text{ and } 1 \le j \le n \tag{10}$$

where, m_c is the mean to which the datum x_j belongs. The following iterative process optimizes the data grouping and the mean values:
Let $x_j(t)$ be an input and $m_c(t)$ the value of m_c at iteration t (t = 0,1,2,…), then

$$m_c(t+1) = m_c(t) + \alpha(t)[\, x_j(t) - m_c(t)] \tag{11}$$

if x_j belongs to the group of m_c, and

$$m_c(t+1) = m_c(t) \tag{12}$$

if x_j does not belong to the group of m_c
$\alpha(t)$ [$0<\alpha(t)<1$] is a monotonically decreasing scalar learning rate. The iterations stop at either after a certain number of cycles decided by the user or when the condition $|m_c(t+1)-m_c(t)| \le \delta$ is satisfied, where δ is an error limit assigned by the user. The variances of membership functions can be determined by Eq.(13) below:

$$\sigma_i = \frac{1}{R}\sqrt{\frac{1}{p_i}\sum_{j=1}^{p_i}(x_j - m_i)^2} \qquad 1 \le i \le k \tag{13}$$

where σ_I = Variance of membership function *i,*

m_I = Mean of membership function *i*,

x_j = Observed data sample,

k = Total number of membership function nodes,

p_I = Total number of data samples in *ith* membership function group, and

R = Overlap parameter.

For a given input or output variable, the number of initial mean values (m_1, m_2, …, m_k) is assigned by trial-and-error. This involves striking a balance between learning time and accuracy. Too small a number results in an oversimplified structure and might therefore adversely affect accuracy. On the other hand, too large a number increases network complexity unnecessarily, resulting in a considerable increase in learning time with very little or no increase in accuracy.

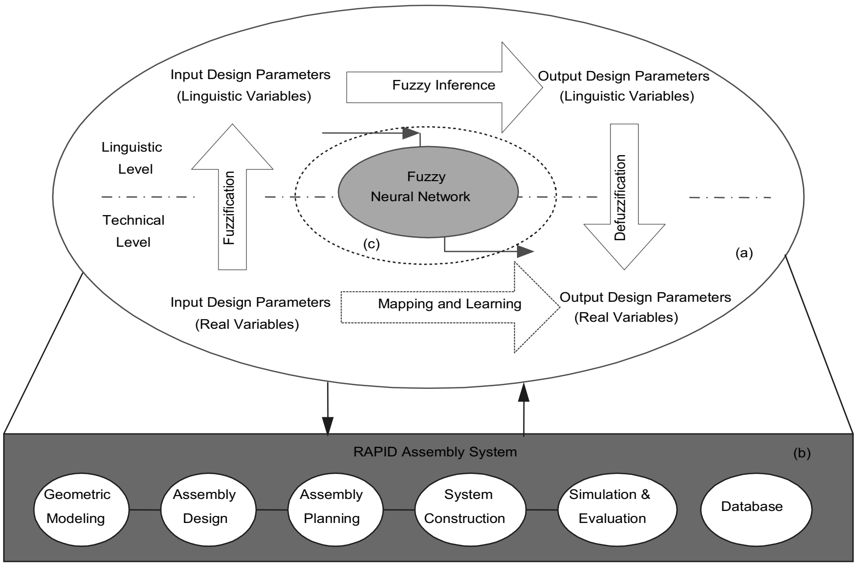


Fig. 3. FNN design system

4. System Implementation

To facilitate the designer to solve the design problems, an FNN-based soft computing design system is developed, as shown in Fig.3. The FNN model has been implemented in computer software using C/C++ in PC Windows's environment. In the real application and implementation of the soft computing framework, the FNN supported soft computing design system is implemented through integrating an FNN mapping system, a fuzzy rule acquisition system, and the RAPID Assembly system (Zha 1999). The fuzzy rule acquisition system is actually a knowledge learning system. The RAPID system is used for virtual prototyping and planning with modules such as geometric modeling, assembly design, assembly sequence planning, assembly system construction, simulation and evaluation, ergonomic database (e.g. anthropometric data), etc. The application examples of FNN using the software are presented below.

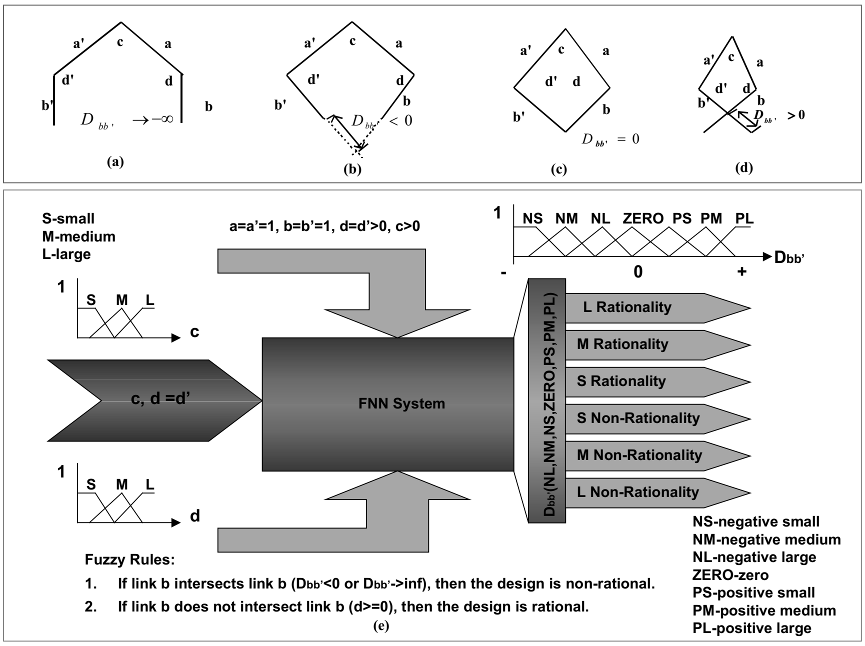
5. Case Study

To verify and validate the developed model and system, a case study is carried out. The design of a virtual four-linkage planar mechanism could be represented by four parameters, i.e., linkage lengths *a*=*a*', *b*=*b*' (unit length), and angles *c*, *d*=*d*' ∈[0,180](degree). The design constraints require that linkage *b* does not intersect linkage *b*. Thus, the design has two patterns. If linkage *b* intersects *b*', the design is non-rational ("1" type), otherwise, it is rational ("0"). In fact, the design problem can be described as follows:

(1) The rational design satisfies either: *b* and *b*' never intersect; or the intersecting point is not on the *b* or *b*' if intersects.

(2) The non-rational design satisfies both: *b* and *b*' intersect; and the intersecting point is on the *b* or *b*' if intersects.

(3) The distance $D_{bb'}$ between the intersecting point and the end point of b or b' can be used to describe the rationality degree of design, e.g., very rational, more rational, less rational, etc. which corresponds to [0,1].



Fuzzy Rules:

1. If link b intersects link b (Dbb'<0 or Dbb'->inf), then the design is non-rational.
2. If link b does not intersect link b (d>=0), then the design is rational.

Fig. 4. Four-linkage mechanism designs with FNN

Fig. 4 (a)-(e) show the rational and non-rational designs respectively. We also can derive some fuzzy rules for the problem, as follows,

Fuzzy Rules:
IF *c* is S (small) and *d* is L (large)
THEN $D_{bb'}$ is negative large (NL) and the rationality degree is positive large (PL).

IF $\quad c$ is S (small) and d is M (medium)
THEN $D_{bb'}$ is positive medium (PM) and the rationality degree is negative large (NL).
……

The training patterns/samples are partially listed in Table 1. After training, the model can evaluate or classify the non-rational and rational designs, and further rectify or modify or re-design the non-rational preliminary designs (e.g., *a*=1, *b*=1, *c*=20, *d*=90) into the rational ones.

Table 1. Training patterns/samples

a	**b**	**c**	**d**	$D_{bb'}$	**Rationality**
1	1	20	90	0.824	NL
1	1	20	170	$-\infty$	PL
1	1	100	90	-0.192	PS
1	1	100	170	-1.192	PL
1	1	20	91	0.823	NL
…	…	…	…		…

6. Conclusions

This paper proposed a soft-computing framework for engineering design and simulation based on the neural and fuzzy hybrid intelligent system techniques. The FNN, which is basically a five-layer fuzzy rule-based neural network, has been developed and integrated with the soft computing design system. The developed fuzzy neural network model can be used as a toolkit for virtual product design. It brings trade-off analysis into the design process, allowing engineers to adopt a "predict and prevent" approach. The developed soft computing design system facilitates the application of the proposed FNN model for design support. Within the system, only one network is needed to train for accomplishing the evaluation and modification tasks. In summary, the fuzzy neural network hybrid system proposed in this paper thus provides a better solution to the problems of virtual product design.

Disclaimer: The work reported here by the author was conducted during his tenure at Singapore Institute of Manufacturing Technology, Singapore. No approval or endorsement by the National Institute of Standards and Technology is intended or implied.

References

Azoff, E. M. (1994), Neural Network Time Series Forecasting of Financial Markets, John Willey & Sons

Bateman, R.E., Bowden, R.G., Gogg, T.J., Harrell, C.R. and Mott, J.R.A. (1997), System Improvement Using Simulation, Fifth Edition, Promodel Corporation

Dagli, C.H. (1994), Artificial Neural Networks for Intelligent Manufacturing, Chapman & Hall.

Goonatilake, S. and Khebbal, S. (1995), Intelligent Hybrid Systems, John-Wiley & Sons

Horikawa, S., Furuhasshi, T. and Uchikawa, Y. (1992), On Fuzzy Modeling Using Fuzzy Neural Networks with the Back-propagation Algorithms, IEEE Transactions on Neural Networks, 3(5), pp.801-806

Jang J-S Roger (1993) ANFIS: Adaptive-network-based Fuzzy Inference System, IEEE Transactions on System Man Cybernetics, 23: 665–685

Kang, S. Y. (1991), An Investigation of the Use of Feedforward Neural Networks for Forecasting, PhD Thesis, Kent State University

Kosko, B. (1992), Neural Networks and Fuzzy System, Prentice-Hall International Editions

Kohonen, T. (1988), Self-organization and Associative Memory, Springer-Verlag, Berlin

Law, A.M. and Kelton, W.D. (1991), Simulation Modeling & Analysis, Second Edition, McGraw-Hill International Editions

Leonard, J.A. and Kramer, M.A., (1991), Improvement of the Back-Propagation Algorithm for Training Neural Networks, Computer Chemical Engineering, 14, pp.337-341

Li, X, Tay, A., and Ang, C.L. (2000), A Fuzzy Neural Network for Virtual Equipment Design, Research Report, SIMTECH, Singapore

Medsker, L.R. (1995), Hybrid Intelligent Systems, Kluwer Academic Publishers.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J.(1986), Learning Internal Representations by Error Propagation, Parallel Distributed Processing, Vol. 1, MIT Press, Cambridge.

Wang, J. and Takefuji, Y. (1993), Neural Networks in Design and Manufacturing, World Scientific, Singapore

Xiang, W., Fok, S.C., and Yap, F.F. (2001), A Fuzzy Neural Network Approach to Model Hydraulic Component from Input–Output Data, International Journal of Fluid Power, 2 (1): 37–47

Zha, X.F. (1999), Knowledge Intensive Methodology for Intelligent Design and Planning of Assemblies, PhD Thesis, Nanyang Technological University, Singapore

Zha, X.F. (2003), Soft Computing Framework for Intelligent Human–Machine System Design, Simulation and Optimization, Soft Computing, 7:184 -198

Zha, X.F. (2004a), A Hybrid Cross-Mapping Neural Network Model for Computational Intelligent Design, International Journal of Knowledge-based and Intelligent Engineering Systems, 8(1): 17-26

Zha, X.F., (2004b), Artificial Intelligence and Integrated Intelligent Systems in Product Design and Development, Intelligent Knowledge-based Systems: Business and Technology in New Millennium, vol. IV: Intelligent Systems, Chapter 1, Cornelius T. Leondes (ed), Kluwer Academic Publishers, USA

Zarefar, H., and Goulding, J.R. (1992) Neural Networks in Design of Products: A Case Study, Intelligent Design and Manufacturing, Kusiak A (ed.), pp. 179–201, John Wiley & Sons, Inc.

Zurada, J. M. (1992), Introduction to Artificial Neural System, West Publishing Company, USA

Internet Server Controller Based Intelligent Maintenance System for Products

Chengliang Liu[1], Xuan F. Zha[1,2], Yubin Miao[1], Jay Lee[3]

1-Institute of Mechatronics Control, Shanghai Jiao Tong University Shanghai 200030, China; 2-National Institute of Standards and Technology, Gaithersburg, MD 20899,U.S.A.; 3-Center for Intelligent Maintenance, University of Wisconsin at Milwaukee Milwaukee, WI 53211, U.S.A

Abstract: This paper presents the development of an Internet server controller based intelligent maintenance system for products. It also discusses on how to develop products and manufacturing systems using Internet-based intelligent technologies and how to ensure the product quality, coordinate the activities, reduce costs and change maintenance practice from breakdown reaction to breakdown prevention. In the paper, an integrated approach using hardware and software agents (watchdog agent) is proposed to develop the Internet server controller based integrated intelligent maintenance system. The effectiveness of the proposed scheme is verified by developing a real system for the washing machine.

Key words: e-Products, e-Manufacturing, e-Service, remote monitoring, diagnosis, maintenance, integrated intelligent systems, multi-agent system, neural computing, Internet, and WWW

1. Introduction

In recent years, web-based e-system technologies have imposed the increasing impact on the evolution "velocity" for product design, manufacturing, and business operations. Business automation is forcing companies to shift operations from the traditional "factory integration" strategy into the "virtual factory" and supply-chain management strategy (NRC 1990). The technological advances to achieve this highly collaborative design and manufacturing environment are based on the multimedia type of information tools and highly reliable communication systems to enable distributed procedures in concurrent design, remote operation of manufacturing processes, and operation of distributed production systems. This transition is dependent upon the advancement of next-generation manufacturing practices on "e-factory and e-automation" focused on the use of information to support collaboration on a global basis. Quality is no longer a unique objective but a prerequisite for competition in the global marketplace. In addition, the complexity of today's products has greatly drawn consumer's attention on the service costs and values of the product's life cycle. A new robust paradigm focusing on e-intelligence for integrated product design, manufacturing, and service has be-come a new benchmark strategy for manufacturing companies to compete in the twenty-first century (Lee 1998, 1999, 2000). A leading manufacturing organization needs to be flexible in management and labor practices, and possess the ability to develop and produce virtually defect-free products quickly (supported with global customer service) in response to opportunities and needs of the changing world market. An e-Intelligence Maintenance System is needed for the next-generation product and manufacturing systems. Future smart manufacturing companies necessitate a set of core intelligence to address the issues of smart business performance within the integrated design, manufacturing, and service business system. This set of core intelligence is called "5Ps," namely predictability, producibility, productivity, pollution prevention, and performance. Scientifically, such kind of integrated business system is subject to the well-established prediction and optimization methods. In this case, with understanding of the manufacturing business, the focus should be on customer's solution and the optimum action should be found out for innovation in the product life cycle.

The main objective of this research is to reduce the downtime of equipment near zero by: 1) developing an Internet-based server controller; 2) embodying a prognostics on-a-chip Watchdog Agent for product behavior assessment and performance degradation; 3) evaluating; 4) establishing a teleservice engineering system testbed (TEST); 5) developing integrated intelligent maintenance software system. This paper reports the development of an Internet server controller based intelligent maintenance system for information appliance products. A hybrid intelligent approach using hardware and software agents (watchdog agent) is proposed to develop the system. The organization of the paper is as follows. Section 2 provides the architecture of an intelligent maintenance system. Section 3 discusses a server controller used for the intelligent maintenance system. Section 4 develops and embodies a prognostics on-a-chip watchdog agent for assessing and evaluating product behavior and performance degradation. Section 5 is the discussion of the TEST and the development toolkit for the integrated intelligent maintenance software system. Section 6 gives some concluding remarks.

2. Intelligent Maintenance Systems

To provide customers with better solutions, a sort of intelligent product service system should be developed, in which a smart software system can predict the failure of a product in advance. Such "prognostic" capability can provide a new kind of aftermarket service to guarantee product and operation performance. Thus, when developing products, the developer should realize that a customer's need should be extended to the whole lifecycle of a product, and this requires that product design should take into account aftermarket support. Service and maintenance are important practices to maintain product and process quality and customer satisfaction.

Currently, many manufacturing companies are still performing service and maintenance activities based on a reactive approach. The fundamental issues for resolving these problems are inadequate information/understanding about the behaviors of products and manufacturing equipment on a daily basis. How to measure the performance degradation of components and machines is still an unsolved issue. Developers also lack the validated predictive models and tools that can tell what would happen when the process parameters take specified values. Research is required to understand the factors involved in product and machine breakdown and to develop smart and reconfigurable monitoring tools that reduce or eliminate the production downtime, and thus reduce the dimensional variation due to the process degradation. To achieve these goals, smart software and NetWare are needed to provide proactive maintenance capabilities such as performance degradation measurement, fault recovery, self-maintenance, and remote diagnostics. These features would allow manufacturing and process industries to develop proactive maintenance strategies to guarantee the product and process performance and ultimately eliminate unnecessary system breakdowns. When aging occurs, the component and machine generally progress through a series of degradation states before failure occurs. Fig.1 shows the architecture of an intelligent maintenance system.

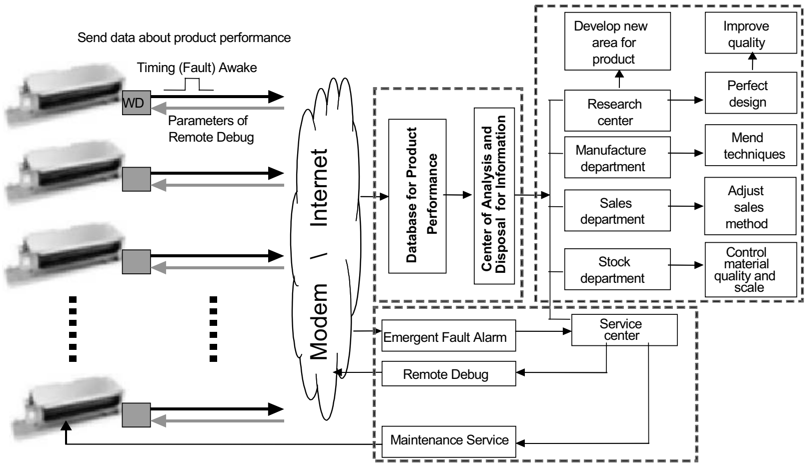


Fig. 1. Intelligent maintenance system

The assessment of machine's performance information requires an integration of many different sensory devices and reasoning agents. The operational performance of components, machines, and processes can be divided into four states: 1) normal operation state, 2) degraded state, 3) maintenance state, and 4) failure state. If a degradation condition is detected, then proactive and corrective maintenance activities can be performed before a worse degradation condition or failure occurs. Fig.2 shows the process of products degradation.

To achieve a just-in-time lean maintenance and service operation, better decision-making tools are required to strategize resources through a sup-ply-chain network system. Currently, many commercially available maintenance management software tools lack integration ability and interoperability with production control systems. They also lack values in managing maintenance logistics and service business. Companies need to perform remote maintenance decision making to support technical and business personnel in manage maintenance and service logistics activities. In addition, digital service modem can be used to integrate product's performance conditions with the customer support center via Internet.

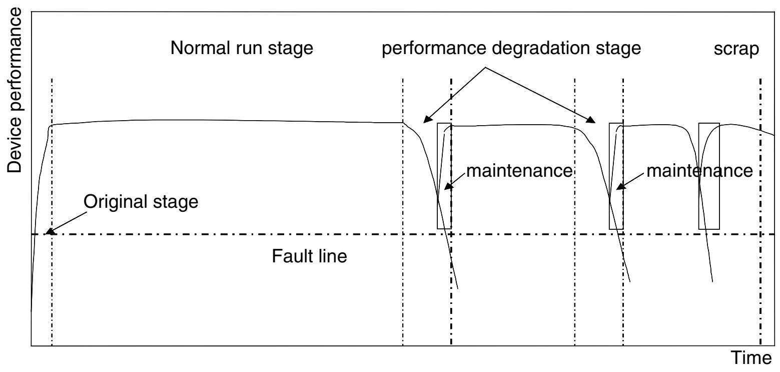


Fig. 2. Device time-performance curve

3. Internet-based Server Controller

3.1. Structure of the Embedded Network Model

The server controller is web-enabled. An embedded network model is the core of the server-controller. Next-generation products are to be a B2D-based (Business-To-Device) global system, in which the server controller will be the base. Through the service controller, the service center can collect performance data and monitor all the products dispersed all over the world at any time. This requires that the networked products have an IP when it’s sold. The service center can also investigate the working states of any products and give the assessment results to the user. The Internet-based server controller has the following functions: 1) It’s a controller. 2) It's a web-enabled server, i.e., it can be connected to

Internet/Intranet. The product can be controlled in the remote site. 3) It can monitor the equipment by collecting and processing the performance data and processing.

Fig.3 shows the structure of the embedded network model composed of nine parts, including CPU, flash, RAM I/O, RJ-45, etc. In this model RTL8019 is used as the network controller and RABBIT2000 is used as the CPU. Fig.4 shows the photo of the server-controller (hardware).

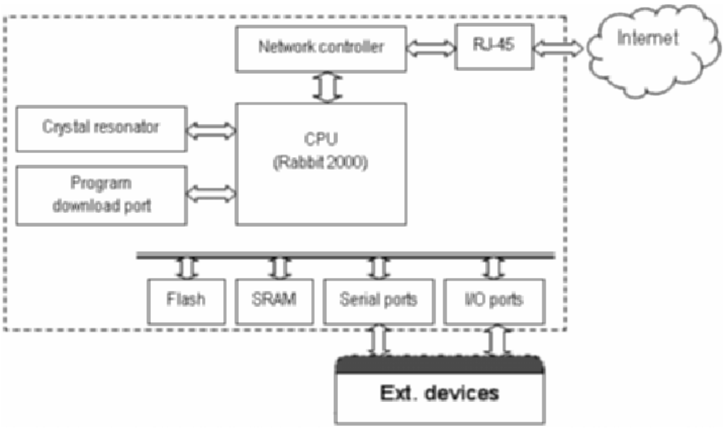


Fig. 3. Structure of Server controller

Fig. 4. The photo of the server-controller (hardware)

3.2. Software Agent for Embedded Network Model

The software of the embedded network model is a software agent in the system. It is used to coordinate and control the embedded network. Fig.5 shows the flowchart of the agent program in the network device in the remote monitoring system. The software agent is actually a package of Smart Prognostics

Algorithms, which consists of embedded computational prognostic algorithms and a software toolbox for predicting degradation of devices and systems. A toolbox that consists of different prognostics tools has been developed for predicting the degradation or performance loss on devices, process, and systems. The algorithms include neural network based, time-series based, wavelet-based and hybrid joint time-frequency methods, etc. (Fig.6). The assessment of performance degradation is accomplished through several modules including processing of multiple sensory inputs, extraction of features relevant to description of product's performance, sensor fusion and performance assessment. Each of these modules is realized in several different ways to facilitate the use of software agent in a wide variety of products and applications. Various requirements and limitations with respect to the characteristics of signals, available processing power, memory and storage capabilities, etc. should be considered.

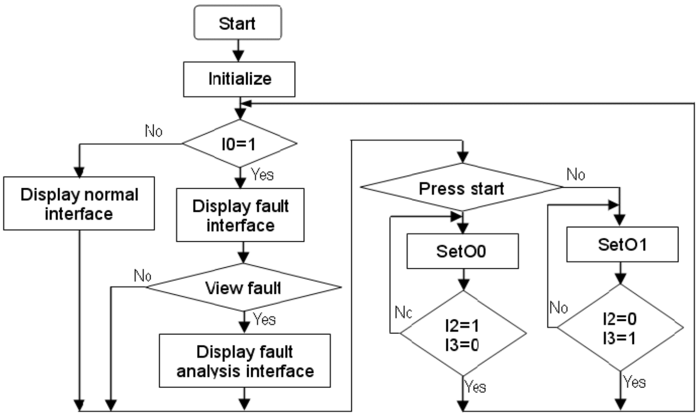


Fig. 5. The flowchart of the agent program for the network device

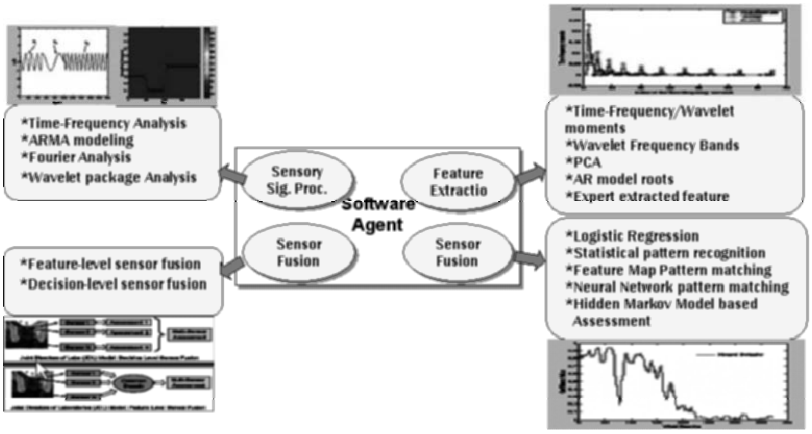

*Time-Frequency Analysis
*ARMA modeling
*Fourier Analysis
*Wavelet package Analysis
Sensory Sig. Proc.
Feature Extractio
Software Agent
Sensor Fusion
Sensor Fusion
*Time-Frequency/Wavelet moments
*Wavelet Frequency Bands
*PCA
*AR model roots
*Expert extracted feature
*Feature-level sensor fusion
*Decision-level sensor fusion
*Logistic Regression
*Statistical pattern recognition
*Feature Map Pattern matching
*Neural Network pattern matching
*Hidden Markov Model based Assessment

Fig. 6. Smart prognostics algorithms of software agent

4. Watchdog Agent

The Watchdog Agent integrated with wireless sensors provides integrated sensor acquisition, processing, hashing, and health (good condition) reporting. The Watchdog Agent performs multi-input data generalization, hashing and mapping, and represents the performance status information on a look-up table using a cerebellum computing approach. Fig.7 shows the computing scheme of the Watchdog Agent.

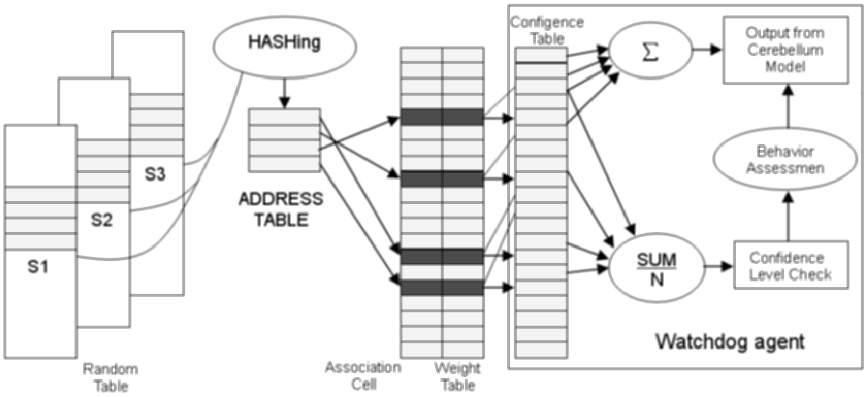


Fig. 7. Computing scheme of the Watchdog Agent

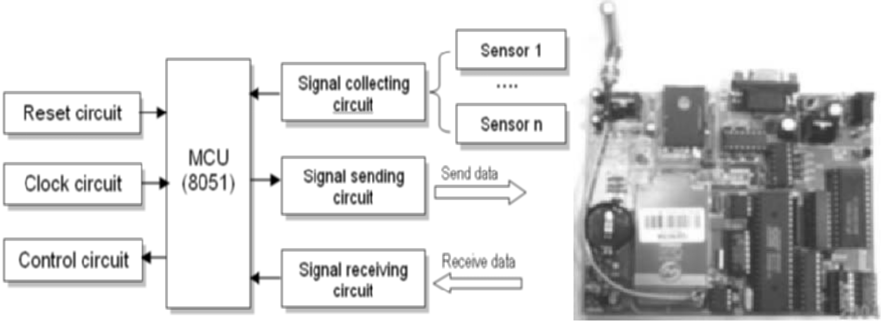


Fig. 8. The simplified hardware diagram of the Watchdog

Fig.8 shows a simplified diagram of the assistant controller. From the diagram we can see that seven parts compose the controller:

1) MCU (8051): the CPU of the controller managing and controlling the actions of every part.
2) Reset circuit: makes 8051 to restore the original state.
3) Clock circuit: provides outside clock for 8051.
4) Control circuit: sends the control signals to start or stop the washing machine.
5) Signal collecting circuit: collects the primary signals through sensors.
6) Signal sending circuit: sends the results first to the web chip then to the monitoring module after the signal is processed in 8051.

7) Signal receiving circuit: receives the control order from the webchip and then sends it to 8051.

5. Tele-Service Engineering System for Information Appliance and Testbed

Information appliances and other interactive "beyond the desktop" products present some user interface design challenges. An information appliance involves a strong connection between the appliance and the network. Besides providing information it embodies the concept of e-home and e-service. Thus, an information appliance is a combo of 3C (communication, computer and consumed electric production). The user can get information over the network, control the operation mode of appliance in remote site, and obtain more services from the factory, not only its products. In a word, the user can obtain more comfort and pleasure. In this section, we discuss a teleservice engineering system testbed for information appliance (consumer appliance).

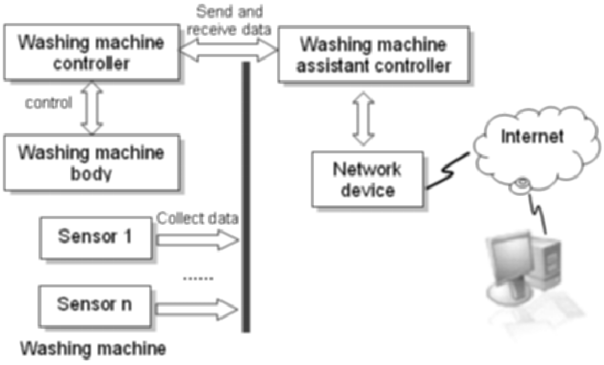
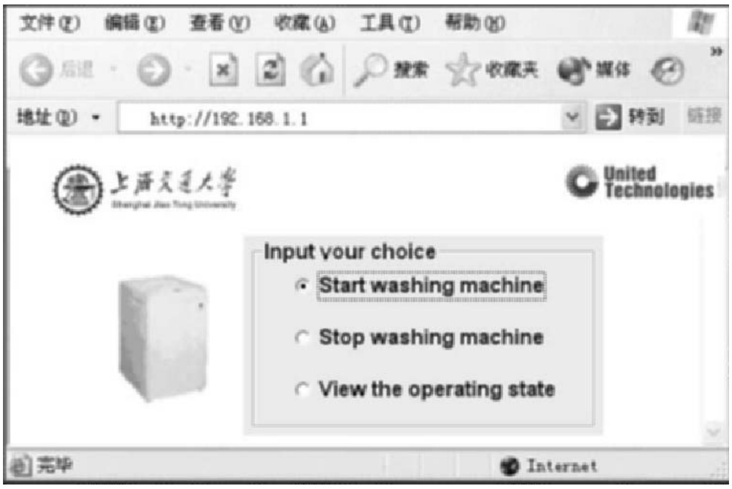


Fig. 9. Structure of the remote engineering system testbed

5.1. Structure of the Remote Engineering System Testbed

We have developed a remote monitoring platform in which many instruments and devices can be monitored. In this paper, a washing machine is selected as an example of the monitoring object due to its universality and practicability. Fig.9 shows the structure of the remote engineering system testbed. The system consists of two parts. The first part is an assistant controller that receives control signals from the network device and controls the operation mode of washing machine. In the meanwhile, it receives sensory information from various sensors and sends to the network device after processing. The second part is the network device developed on the base of a web chip. This part acts as the tool that communicates with monitoring PC through Internet.



Input your choice

● Start washing machine

○ Stop washing machine

○ View the operating state

Fig. 10. Remote Engineering System Testbed for Information Appliance

5.2. Main Interfaces of the Remote Engineering System Testbed

Fig.10 shows a remote engineering system testbed for the washing machine. There are four buttons in this interface. "Start the washing machine" and "Stop the washing machine" can control the operation state of the washing machine. The user can view the running state of the washing machine through pressing the button "View the operation state of washing machine". The last button "Home page" can make the user returning to the main homepage of the system.

5.3. The Development Toolkit

In this work Dynamic C is used as the development toolkit. Dynamic C is an integrated development system for writing embedded software for Rabbit microprocessors. It provides many function libraries, all in source code. These libraries can help the user/customer to write the most reliable embedded control software.

6. Conclusions

This paper presented the development of an Internet server controller based intelligent maintenance system for information appliance products (washing machine). It also discussed on how to develop products and manufacturing systems using Internet-based intelligent technologies and how to ensure product quality, coordinate activities, reduce costs and change maintenance practice from the breakdown reaction to prevention. The Internet server controller based intelligent maintenance system was developed using hardware and software agents (watchdog agent). A real intelligent maintenance system for the washing machine was developed to verify the effectiveness of the proposed approach.

Acknowledgement and Disclaimer

We are happy with the help and support of Mr. Weiping Wu and Dr. Guoquan Zhao from UTRC. They give us the collaborating opportunities. This work is supported by the National Natural Science Foundation of China (Grant No.50128504). No approval or endorsement by the US National Institute of Standards and Technology is intended or implied.

References

NRC (1990), National Research Council, The Competitiveness Edge: Research Priorities for U.S. Manufacturing, National Academy Press.

Jack Welch and G E. (1996), Business Week, Oct. Issue

Lee, J. and Kramer, B.M. (1992), Analysis of Machine Degradation Using a Neural Network Based Pattern Discrimination Model, J. Manufacturing Systems, Vol. 12, No. 3, pp. 379-387

Shi, J., Ni, J., and Lee, J. (1997), Research Challenges and Opportunities in Remote Diagnostics and System Performance Assessment, Proceeding of 4th IFAC Workshop on Intelligent Manufacturing Systems: IMS'97, July 21-23, Seoul, Korea.

Lee, J. (1998), Teleservice Engineering in Manufacturing: Challenges and Opportunities, International Journal of Machine Tools & Manufacture, 38, p901-910.

Lee, J. (1999), Machine Performance Assessment Methodology and Advanced Service Technologies, Report of Fourth Annual Symposium on Frontiers of Engineering, National Academy Press, pp.75-83, Washington, DC.

Lee, J., and Wang, B. (1999), Computer-aided Maintenance: Methodologies and Practices, Kluwer Academic Publisher.

Lee, J. (2000), E-Intelligence for Product and Manufacturing Innovation, Invited Special Lecture Paper, Proceeding of 7th CIRP International Symposium on Life Cycle Engineering, Tokyo

Botkin, Jim. (1999), Smart Business, The Free Press

Kurzweil, Ray (1999), The Age of Spiritual Machine, Penguin Books.

Charan, R. and Tichy, N. (1998), Every Business is a Growth Business, Times Business.

A Novel Genetic Fuzzy/Knowledge Petri Net Model and Its Applications

Xuan F Zha[*]

National Institute of Standards and Technology, USA

Abstract. Every individual intelligent technique has particular computational properties (e.g. ability to learn, explanation of decisions) that make it suited for particular problems and not for others. There is now a growing realization in the intelligent systems community that many complex problems require hybrid solutions. This paper presents a novel genetic (fuzzy) knowledge Petri net based approach for the integration of KBSs, fuzzy logic (FL) and GAs. Two genetic Petri net models are presented for integrating genetic models and knowledge-based models, including genetic knowledge Petri nets and genetic fuzzy knowledge Petri nets (GKPN, GFKPN). The GKPN and GFKPN models can be used for (fuzzy) knowledge representation and reasoning, especially for (fuzzy) knowledge base tuning and verification & validation. An application example for the proposed models in engineering design is given. .

Keywords: Knowledge Petri net, fuzzy logic, fuzzy knowledge Petri net, knowledge-based systems, genetic algorithms, genetic programming, and genetic fuzzy/knowledge Petri net

1. Introduction

To date, very little work has been reported on the model integration of expert systems with genetic algorithms. Genetic techniques have been used to improve the performance of rule-based systems and related problem-solving techniques in the artificial intelligence and operations research fields. Recently, however, a more complicated but promising way has been considered as the implementation of GAs within a knowledge-based environment. Along this way, one of the most exciting approaches is using Petri net as a unified model to implement GAs within a knowledge-based environment. This is because Petri net, as an important analytical and modeling tool for the discreet event dynamic system (Peterson 1981), has the capacity for the modeling, representation and analysis. Its place

[*] Email: zha@cme.nist.gov

accumulates tokens and its output transition fires when the sum of tokens exceeds a threshold. The places and transitions are interconnected in various ways to provide parallelism, synchronism and other properties. On the other hand, Petri net possesses the potentials to be incorporated into artificial intelligence framework (KBS) with knowledge representation, automated reasoning and decision-making (Generich and Tieler-Mevissen 1976, Generich and Lautenbach 1981). Based on the combination of knowledge processing and the basic place-transition Petri nets, various classes of extended Petri nets such as fuzzy Petri net, knowledge Petri nets, fuzzy knowledge Petri nets, object-oriented intelligent Petri nets, have been formalized (Alberto & Barro 1994; Looney 1988; Zha et al. 1997a,b, Zha et al. 1998). Consequently, a number of KBSs have been built upon these extended Petri nets. Looking on the relationships between Petri net and KBS, GAs and KBS, it is therefore possible to integrate KBSs and GAs through Petri net.

This paper first identifies and describes how knowledge based systems (KBSs), fuzzy logic (FL) and genetic algorithms can be integrated, and then it provides a novel genetic (fuzzy) knowledge Petri net-based approach for the integration of KBSs and GAs. This paper is focused on two genetic Petri net models for integrating genetic models and knowledge-based models, including genetic knowledge Petri nets and genetic fuzzy/ knowledge Petri nets (GKPN, GFKPN). The remaining parts of this paper are organized as follows. Section 2 proposes knowledge Petri net models. Section 3 proposes two novel genetic knowledge Petri net models (GKPN, GFKPN). Section 4 presents the application and case study using the developed model and system. Section 5 summarizes the paper.

2. Knowledge-Based Petri Net Models

Petri net possesses the potentials to be incorporated into artificial intelligence framework with knowledge representation, automated reasoning and decision-making. Knowledge Petri net (KPN) may be built upon the extensions of the basic place-transition Petri nets and incorporation with knowledge-based expert system (Zha et al. 1997 a,b).

2.1 Knowledge Petri Net

The definition of knowledge Petri net can be described as follows (Zha et al 1997, 1998):

Definition 1: Knowledge Petri net is defined as a 7-tuple KPN=(P, T, A, W, M_0, K_P, K_T). P,T and A are similarly defined as that for usual place-transition Petri nets. The places are decomposed into two subsets: P= {NORMAL, CONTROL}, which are a set of normal places similar to the usual Petri net and a set of flow-control places (P_c) respectively. The transitions are of two types: T={NORMAL, TRANSFER_THRESHOLDING}, which are a set of normal transitions similar to a usual Petri net and a set of thresholding transitions (T_d) respectively. W→{-1,0,1} is a weight function on arcs, where, "1" represents an excitatory or permitted arc, while "-1" and "0" represent control-flow (C_f) arcs (inhibitor arcs and non-selective arcs) respectively. M_0: P→{0,1,2,…} is the initial marking. K_P: P→ S_P is a one-to-one mapping from place set P to predicate set S_P. K_T: T→S_R is a one-to-one mapping from transition set T to rule set S_R.

According to the definition of KPN, it can be regarded as a combination of two aspects: net graph (i.e., KPN graph) and knowledge annotations. Similar to usual PTPN graph, KPN graph is a graphic representation of knowledge-based Petri net structure and visualizes the reasoning rules. The steps which transform an PTPN graph into a KPN graph was described as (Zha et al. 1997, 1998). For transforming PN graph to KPN graph: (1) Each place in PN graph corresponds to a place of KPN; (2) Each transition in PN graph corresponds to a transition of KPN; (3) *m* transitions with one output place correspond to *m* transitions, *m*-1 associated flow-control places and 2(*m*-1) associated permitted and/or inhibited arcs of KPN (*m*>1).

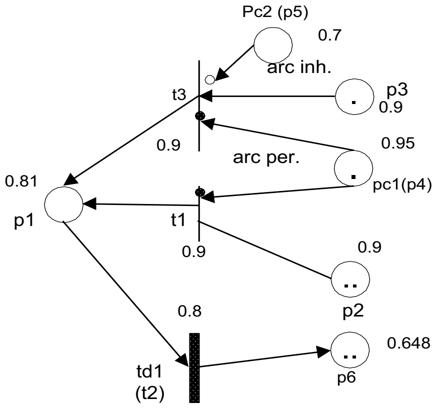
The knowledge annotations are composed of the place knowledge annotations and the transition knowledge annotations, i.e., K_P and K_T, which enlarge the representation ranges and contents of its net graph. The knowledge annotations can be organized into a knowledge base, and KPN is therefore a knowledge-based expert system. Other characteristics of knowledge-based Petri net are similar to the conventional Petri nets and given in (Zha et al. 1998a).

2.2 Fuzzy Knowledge Petri net

The integration of fuzzy logic and expert system leads to the natural extension of fuzzy expert systems. A fuzzy knowledge-based Petri net (FKPN) may be built upon the knowledge-based Petri net incorporated with fuzzy logic.

Definition 2: Fuzzy knowledge-based Petri net is defined as a 7-tuple FKBPN=(P, T, A, W, M_0, K_{FP}, K_{FT}). P, T and A are defined as that for KPN. The places are decomposed into two subsets: P= {NORMAL, CONTROL}, which are a set of normal places similar to the usual fuzzy Petri net (Albert & Barro 1994, Ahson 1995) and a set of flow-control places respectively. The transitions are of two types, T= {NORMAL, TRANSFER}, which are a set of normal transitions similar to the usual fuzzy Petri net and a set of fuzzy transfer transitions (e.g., Gaussian function: y=$e^{-1.0536x}$, *x* is the input and *y* is the output of possibility, Zarefar and Goulding 1991) respectively. W→ [0,1] is a fuzzy weight function on arcs. M_0: P→ {0,1} is the initial marking. K_{FP}: P→ $S_{FP}\cup f_{\mathbf{P}}$ is a one-to-one mapping from place set P to the union set of the linguistic variable and predicate set S_{FP} and the fuzzy membership set f_P . K_{FT}: T→ $S_{FR} \cup f_{\mathbf{T}}$ is a one-to-one mapping from transition set T to the union set of fuzzy rule set S_{FR} and certainty factor set f_T. $f_{\mathbf{P}}$: P→[0,1] is a set of fuzzy memberships of places, i.e., $f_{\mathbf{P}}$

$=\{ f_{p_i}$,i=1,2,…,m}. f_T: T→[0,1] is a set of certainty factors of transitions, i.e., f_T $=\{ f_{t_j}$, j=1,2,…,n }. According to the formation and definition of FKPN, it can be regarded as the fuzzification of KPN using fuzzy logic. Therefore, KPN is a special case of FKPN. An illustration of fuzzy knowledge Petri net graph is shown in Fig.1 below.



P={NORMAL, CONTROL)={{p_1, p_2, p_3, p_6}, {p_{c1}, p_{c2}}}; T={NORMAL, TRANSFER)={{t_1, t_3}, {t_{d1}}}; A={α,β}={{ (p_1, t_{d1}), (p_2,t_1), (p_3, t_3), (p_{c1},t_1), (p_{c1}, t_3), (p_{c2},t_3)}, {(t_1, p_1), (t_{d1},p_6), (t_3,p_1), (t_{d1}, p_6)}}; M_0= $(0,2,1,1,0,0)^T$; C_f: (p_{c1},p_1), (p_{c1},t_3) are permitted arcs, marked arc_{per}, (p_{c2},t_3) is inhibited arcs, marked $arc_{inh.}$, and (p_2,t_1) is nonselective arc; W={1,0,1,1,1,-1,1,1,1}; $f_P=\{f_{p1},f_{p2},f_{p3},f_{p4},f_{p5},f_{p6}\}$ ={0.81,0.9,0.9,0.95,0.7,0.648}; $f_T=\{f_{t1},f_{t2},f_{t3}\}$={0.9,0.8,0.9}.

Fig. 1. KPN graph

2.3 FKPN-based Expert System

2.3.1 Knowledge Representation

In FKPN based knowledge representation, three types of knowledge are represented: 1) Descriptive and fact knowledge can be represented by knowledge annotations of places; 2) Rule knowledge can be represented by corresponding knowledge annotations of transitions; and 3) Control knowledge can be represented by flow-controls. These are meta-rules that select one of conflicting transitions as a next firing transition in a conflict resolution. The knowledge such as facts, rules (meta-rules) and flow-control can be represented by predicate logic, attributes list, semantic network, and if-then production rule in AI. Three types of knowledge represented by FKPN can be organized into knowledge base and inference engine of an expert system.

A production rule may be divided into the parts of precondition and conclusion to describe the causality. Let *x* represents the precondition and *y* represents the conclusion. The general form of a production rule is **R**: IF x THEN y. To describe uncertain knowledge, a fuzzy production rule may be represented as **FR**: IF x THEN y [CF (R)], where, CF(R) $\in$ [0,1] is a certainty factor of the rule which describes the truth-value of the knowledge. This rule shows that if x is true, the conclusion y is true, but the certainty is CF(R). The precondition x and conclusion y may be single fuzzy variables or compound conditionals with multiple sub-preconditions combined with logical conjunction, disjunction or negation. The precondition x may include many sub-preconditions, i.e., x= (x_1, x_2, …, x_n) and may be either "AND" or "OR" relationships of these sub-preconditions. If "AND", the certainty factor of x is the minimum value among all certainty factors of sub-preconditions, i.e., CF(x)= min {CF(x_1), CF(x_2),…,CF(x_n)}. If "OR", the certainty factor of y is the maximum value among all certainty factors of sub-preconditions, i.e., CF(x)=max{ CF(x_1), CF(x_2),…, CF(x_n)}. Likewise precondition x, the conclusion y may be obtained from many fuzzy production rules, its certainty factor CF(y) is as follow: CF(y) = max {CF(y_1), CF(y_2),…, CF(y_{m1})}, where CF(y_1), CF(y_2),…, CF(y_{m1}) are certainty factors of the conclusions which are respectively obtained from every rules. Petri nets have been used for the representation of such clausal form rules and for query processing through the use of invariants (Ahson 1995).

Fig.2(a)(b) shows the following two fuzzy production rules: a) IF x THEN y [CF (R)]; and b) IF x_1 and x_2 THEN y_1 and y_2 [CF (R)]. Fig.2(c) shows a set of three rules (rule-base): IF (x_1 and x_2) THEN y_1 [CF(R1)]; IF (x_2 and x_3) THEN y_2 [CF(R2)]; and If (y_1 and y_2) THEN g_1 [CF(R3)].

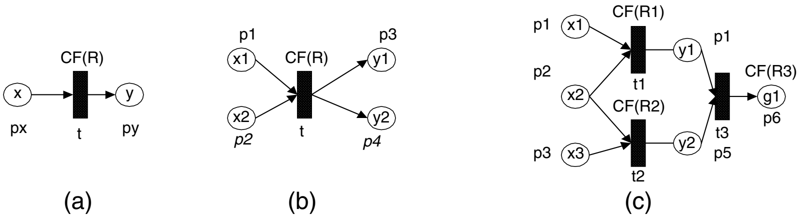


Fig. 2. KPN graph of rules

2.3.2 Reasoning

A hybrid of two search and control strategies is used: concurrent and asynchronous event dispatching method and continuous transition scanning method for problem solving in KPN and FKPN based system. It is an event-driven algorithm. Each event can fire the Petri net asynchronously. The entire system can be carried out concurrently. After an event fires, the Petri net continuously scan the places and transitions to check whether the firing conditions of each transition are satisfied. If the firing conditions of a transition are satisfied, it will fire and tokens of the system will change. This continues until there are no more transitions to be fired. With Petri net graph constructed from the production rules and firing rules of transitions, knowledge reasoning can be carried out and simulated by running the Petri net graph (Zha et al. 1997, 1998).

3. Genetic Knowledge Petri Net Models

Knowledge based (expert) systems represent a logical, symbolic approach while genetic algorithms use numeric and evolutionary processing to mimic models of biological genetic systems. The KPN and FKPN based systems may be combined with GAs to form an even more powerful system.

3.1 Genetic Models for KPN and FKPN

Based on the genetic knowledge in knowledge Petri net model, the genetic model for knowledge Petri net called genetic knowledge Petri net (GKPN) or simply genetic Petri net may be defined.

Definition 3: Genetic Petri net is defined as a 7-tuple GKPN=(P,T, A, W, Mo, K_{GP}, K_{GT}). P,T, A and W are defined as that KPN. K_{GP}: $P \rightarrow S_C$ is a one-to-one mapping from the place set P to the set of chromosomes (usually represented as strings), S_C. K_{GT}: $T \rightarrow S_{GO} = S_R \cup S_{CR} \cup S_{MT}$ is a one-to-one mapping from the transition set T to the union set of genetic operators (crossover and mutation), S_{CR} and S_{MT}, and the set of rule set S_R. Similarly, from the definition of the fuzzy knowledge Petri net and the genetic Petri net (GPN), a new class of Petri nets called genetic fuzzy knowledge Petri nets (GFKPN) or simply genetic fuzzy knowledge Petri net (GFPN) may be defined.

Definition 5: Genetic fuzzy knowledge Petri net is defined as a 7-tuple GFPN=(P,T, A, W, Mo, K_{GFP}, K_{GFT}). P, T, A and W are defined as that FKPN. K_{GFP}: $P \rightarrow S_{FP} \cup S_C \cup S_{f_P}$ is a one-to-one mapping from the place set P to the union set of the linguistic variable and predicate set S_{FP}, the set of chromosomes S_C, and the fuzzy place membership set S_{f_P}. K_{GFT}: $T \rightarrow S_{FCR} \cup S_{FMT} \cup S_{FR} \cup S_{f_T}$ is a one-to-one mapping from the transition set T to the union set of fuzzy genetic operators (fuzzy crossover and fuzzy mutation), S_{FCR} and S_{FMT}, the set of fuzzy rule set S_{FR} and the certainty factor set S_{f_T}.

3.2 Evolutionary Design for Petri Nets

Petri net models here are referred to as from usual place/transition Petri nets to various extended Petri nets such as (fuzzy) knowledge-based Petri nets (KPN, FKPN). The design of Petri net models for specific applications, under given sets of design constraints is to large extent, a process of trial and error, relying mostly on past experience with similar applications. Furthermore, the performance (and cost) of Petri net models on particular problems is critically dependent, among other things, on the choice of primitives (places and transitions), network architecture (e.g. topology and type) and the algorithms. These factors make the process of Petri net model design difficult. In addition, the lack of sound design principles constitutes a major hurdle in the development of large scale Petri nets models and systems for wide variety of practical problems. Motivated by this, it is therefore necessary to investigate efficient approaches for the design of Petri net models.

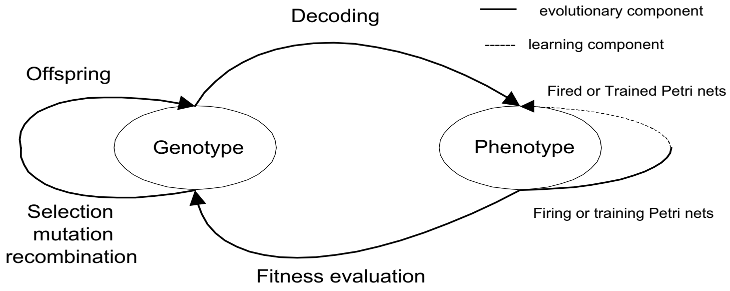

Decoding
—— evolutionary component
------ learning component
Offspring
Fired or Trained Petri nets
Genotype
Phenotype
Selection
mutation
recombination
Firing or training Petri nets
Fitness evaluation

Fig. 3. Evolutionary (genetic) design of Petri nets

The design of Petri net architecture presents us with a challenging multi-criterion optimization problem for particular classes of applications (e.g. modeling, planning, and simulation of discreet event dynamic system, visual pattern classification, inductive and deductive inference, etc.). Evolutionary (genetic) algorithms (Goldberg 1989) offer an attractive and relatively efficient, randomized opportunistic approach to search for near-optimal solutions. The key processes in the evolutionary (genetic) approach to the design of Petri net models are shown in Fig.3.

3.3 Genetic Rule-Finding and -Tuning for FKPN

Genetic techniques can be used to find or tune rules for an expert system, and expert systems can provide heuristics for improving the performance of a genetic system. For example, genetic techniques can be applied to expert networks to find better membership values or certainty factors, and they can be used to optimize the parameters in problem-solving techniques. For FKPN models and systems, genetic algorithms may therefore be used for synthesizing or rule-finding and tuning K_{FP}, and K_{FT}. This is a direct way for the integration of FKPN with genetic algorithms.

In synthesizing fuzzy knowledge-based Petri nets, three features in K_{FP} and K_{FT} may be adjusted: the membership functions defined over the input and output variables, the rule base, and the weights connecting rules to the outputs. Suppose that a FKPN is coded as a single chromosome. The rules are packed end-to-end to form the chromosome. Then a simple GA (Goldberg 1989) can be used to evolve a population of chromosomes, and yield a most-fit chromosome (a FKPN), which represents the relationship between the output and input fuzzy variables in K_{FP}, and $K_{FT.}$ Fixed or static rules representing a priori knowledge are excluded from manipulation by GA operators (crossover, mutation) but are included in the evaluation of a chromosome's fitness. In the simple GA, the length of the chromosome is fixed and thus the number of rules packed in a chromosome is fixed. Therefore, the number of rules in the FKPN must be specified prior to training. A trail-and -error procedure can be used to estimate the number of rules required by running the GA several times with an arbitrarily large number of rules and examining the number of duplicated rules and cut connections. If the FKPN is under saturated (i.e., contains disconnected or duplicated rules), then a number of rules are decreased.

4. Applications in Engineering Design

To verify and validate the developed GFKPN model, the case of designing a simple four-linkage planar mechanism in (Zha 2005b) is used. Fig.4 shows a 5-input-1-ouput GFKPN. After training, the model can evaluate or classify the non-rational and rational designs, and further rectify or modify or re-design the non-rational preliminary designs (e.g., *a*=1, *b*=1, *c*=20, *d*=90) into the rational ones. After evaluation, the non-rational design is to be modified into a rational one by using the Hopfield net (Zha 2004). Fig.5 shows design process of using GFKPN to classify and modify the non-rational preliminary designs (e.g., a=1, b=1, c=20, d=90) into rational ones.

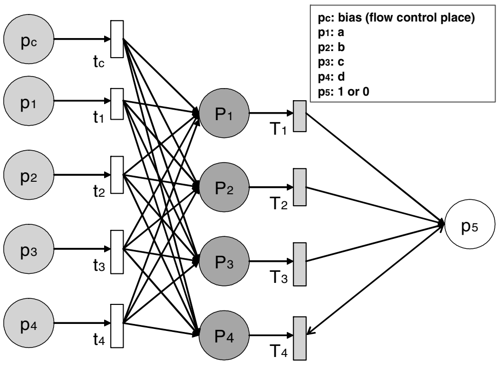


Fig. 4. GFKPN for design evaluation

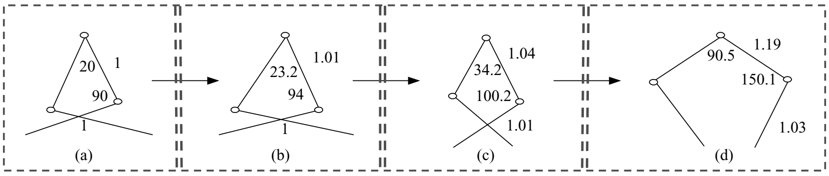


Fig. 5. Redesign process using GFKPN

5. Conclusions

This paper presented a novel unified approach to implementing the integration of knowledge-based systems (KBSs), fuzzy logic (FL) and genetic algorithms (GAs). Two genetic Petri net models were presented for integrating genetic models and knowledge-based models, including genetic knowledge Petri nets and genetic fuzzy knowledge Petri nets (GKPN, GFKPN). The GKPN and GFKPN models can be used for (fuzzy) knowledge representation and reasoning, especially for (fuzzy) knowledge base tuning and verification & validation. With the proposed approach, a unified, knowledge and computation based integrated intelligent system can be achieved.

Disclaimer: The bulk of the work reported here by the author was conducted during his tenure at Nanyang Technological University, Singapore. No approval or endorsement by the National Institute of Standards and Technology is intended or implied.

References

Ahson, S. I. (1995), Petri Net Models of Fuzzy Neural Networks, IEEE Transactions on Systems, Man, and Cybernetics, 25(6): 926-932

Bugarin, A. J. and Barro, S. (1994), Fuzzy Reasoning Supported by Petri Nets, IEEE Transactions on Fuzzy Systems, 2(2): 135-150

Kwasny, S.C. and Faisal, K.A., Rule-based Training of Neural Networks, Journal of Expert System Application, 2(1), 47-58

Chan, S-M., Ke, J.S., and Chang, J.P. (1990), Knowledge Representation Using Fuzzy Petri Net, IEEE Transaction on Knowledge and Data Engineering, 2: 311-319

Garg, M.L., Ahson, S.I. and Gupta, P.V. (1991), A Fuzzy Petri Net for Knowledge Representation and Reasoning, Information Processing Letter, 39: 165-171

Genrich, H.J. and Thieler-Mevissen G. (1976), The Calcus of Facts, Lecture Notes in Computer Science, 45: 588-595, Berlin: Springer-Verlag

Goldberg, D. (1989), Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley

Goonatilake, S. and Khebbal, S. (1995), Intelligent Hybrid Systems, John-Wiley & Sons

Looney, C.G. (1988), Fuzzy Petri Net for Rule-based Decision Making, IEEE Transactions on Systems, Man, and Cybernetics, 14:178-183

Medsker, L.R. (1995), Hybrid Intelligent Systems, Kluwer Academic Publishers

Moore, K.E. and Gupta, S.M. (1996), Petri Net Models of Flexible and Automated Manufacturing Systems: A Survey, International Journal of Production Research, 34(11): 161-170

Nabil Kartam, Ian Flood, and Tanit Tongthong (1995), Integrating Knowledge-based Systems and Artificial Neural Networks for Engineering, Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 9:13-22

Peterson, J.L. (1981), Petri Nets Theory and Modeling of Systems, Englewood Cliffs, NJ: Prentice-Hall

Powell, D.J., Skolnick, M.M. and Tong, S.S. (1995), A Unified Approach for Engineering Design, Intelligent Hybrid Systems, Goonatillake and Khebbal (eds.), pp.107-120

Roston,G. P. (1994), A Genetic Methodology for Configuration Design, PhD Dissertation, Carnegie Mellon University.

Zarefar, H, and Goulding, J.R. (1992), Neural Networks in Design of Products: A Case Study, Intelligent Design and Manufacturing, Kusiak A (ed.), pp. 179–201, John Wiley & Sons, Inc.

Zha, X.F., Lim, S.Y.E., and Fok, S.C. (1997a), Concurrent Intelligent Design and Assembly Planning: Object-oriented Intelligent Petri Nets Approach, Proceedings of the 1997 IEEE Conference on Systems, Man, and Cybernetics (SMC'97, USA)

Zha, X.F. (1997b), Integrated Intelligent Design and Assembly Planning, Research Report, School of Mechanical and Production Engineering, Nanyang Technological University, 255pages

Zha,X.F., Lim, S.Y.E., and Fok, S.C. (1998a), Integrated Knowledge Petri Net based Intelligent Flexible Assembly Planning, Journal of Intelligent Manufacturing, 9: 235-250

Zha, X.F., Lim, S.Y.E., and Fok, S.C. (1998b), Integration of Knowledge-based Systems and Neural Networks: Neuro-expert Petri Net Models and Applications, Proceedings of the 1998 IEEE International Conference on Robotics and Automation, Belgium, pp.1423-1428, Belgium

Zha, X.F. (2004), A Hybrid Cross-mapping Neural Network Model for Computational Intelligent Design, International Journal of Knowledge-based and Intelligent Engineering Systems, 8(1): 17-26

Zha, X.F. (2005a), Soft Computing in Engineering Design: A Hybrid Dual Cross Mapping Neural Network Model, Neural Computing and Applications, 14(3): 176-188

Zha, X.F. (2005b), Soft Computing in Engineering Design: A Fuzzy Neural Network for Virtual Product Design, Soft Computing in Industrial Applications, Advances in Soft Computing Series, Springer Verlag, Germany

Individual Product Customization Based On Multi-agent Technology*

Yong Zhao[1], Cheng Zhao[2], Jianzhong Cha[1]

[1]School of Mechanical and Electronical Control Engineering, Beijing Jiaotong University, Beijing, 100044, China

[2]Department of Mathematics and Computer Science, Indiana State University, Terre Haute, IN 47809, U.S.A.

Abstract: With the increase of global marketing competition and the dynamic change of the marketing environment, products produced by enterprises that are needed by consumers are more individualized and more flexible. It is an important research project for product consumers and enterprises of how to gain product information effectively. Intelligent customization technology of individual products in remote locations is one of the important sides of this project. In this paper, a customized motorcycle is taken as an example to study individual product specifications. A product customization agent is designed to match the contents and to calculate the methods of product value itemized. Using the agents, the traditional delivery of products through addresses can be replaced by product customization communization between consumers and enterprises. Furthermore a calculation method of individual motorcycle customization is proposed.

Keywords: intelligent agent, individual product, system development, motorcycle customization

1 Introduction

With the development of computer/internet technology, the approach of gaining information has been greatly improved. But for enterprises, it is extremely important to send out product information quickly to customers, and to manufacture high qualified products rapidly according to the requirement of customers. It is also important for consumers to gain necessary information of products on the Internet more effectively and accurately. Wupping J. (2000) pointed out that the consumer demands were considered in the process of product

* This work was funded by the NSFC under the grant 50335040.

development and these demands were evaluated in the process of purchasing factors. Although the lack of information of enterprises and consumers has been greatly resolved to a certain extent because of the popularization of network technology, excessive information posted on the network gives consumers a very hard time to find necessary information about the product that is wanted. In order to change this situation, search engines were developed. The search engines can gather concerned network addresses and classify them into different themes. Consumers can make lists from these search engines according to their interests, but this still does not satisfy an effective, quick, easy search. Sometimes the products being searched belong to different themes or the themes that the products belonged to are unknown to that search engine. In that case, the classified themes have no meaning in search engines. Often, when searching a theme word on the network, the searcher can achieve thousands of results but none that leads to what the searcher wants.

As the Internet-based business increases, naturally an agent has been adopted to perform tedious and routine tasks instead of the user or principal (Zwass, V. 1991). An agent is defined as a software program, which performs a given function automatically or semi-automatically by communicating with other computer agents, programs or human agents (O'Leary, D.E.D., Plant, R. 1997). The advantage of an agent is that it acts on behalf of their human users to perform laborious tasks. These tasks include locating and accessing information from various online sources, resolving inconsistencies in a retrieved manner, filtering irrelevant or unwanted information, integrating information from heterogeneous information sources and adapting to human users' needs (Sycara, K., Pannu, A., Williamson, M., Zeng, D. 1996). In an agent, the external communication protocol takes hypertext text markup language (HTML) format, and the internal message handling requires standard generalized markup language (SGML) for document exchange. This agent has the capability to acquire information, to optimize the utilization of resources, and to perform a difficult task by reacting independently and promptly on the changing environment.

The key idea to individual product customization is the concept of negotiation. Negotiation is a discussion in which the interested parties exchange information and come to an agreement. The process behind this idea is as follows. First, there is a two-way exchange of information. Second, each party evaluates the information from their point of view. Third, the contractor submits a proposal, and if both sides agree with the condition, then final decision is made.

So far, multi-agent systems are the best alternatives for characterizing or designing distributed problems. Agents communicate with each other in order to achieve better goals. Communication enables agents to coordinate their actions and behaviors, resulting in systems that are more coherent. A property in a system of agents is coordination which performs some activities in a shared environment. Cooperation is coordination among friendly agents, while negotiation is coordination among competitive or simply self-interested agents. The agents must be able to determine goals shared with other agents, recognize common tasks, avoid unnecessary conflicts, and collect knowledge and evidence (Huhns, M. N., Stephens, L. M. 1999).

KL-ONE is a good knowledge expression system developed by Ronald J. Brachman etc. The description method of object-oriented is taken and conceptions are defined by means of framework format in the knowledge system. The conceptions can be classes and relations, and have a set of characteristics which link slots in the framework with database fields through 'source' side face in order to make use of database mechanism to store and process information. The definition of side face is following.

<Side face value> :: = default | type | mode | number | derive | restriction | unit |source.

Concerning about communication language, a set of format protocol is provided for exchanging information effectively between users and enterprises. KQML (Finin, T., Weber, J. 1993, 1994) is recognized as a communication standard among agents.

In section 2, we study customization principle of individual product. System development and application research are discussed in section 3 and section 4 respectively. Section 5 gives research results.

2 Customization Principles of Individual Product

2.1 Module of Individual Product Customization

In order to carry out an individual product customization, some special information of users must be kept in Web server:

(1) The important category information: Each user has different preferences about products. We use the degree of importance for the category information to represent the interesting grade of some kinds of products.

(2) The historical information of users: According to the historical information of users, the system can recognize some preferences about a product and use a feedback algorithm to adjust the interesting grade of users for some products. For example, if a user likes a product in red, then the interesting grade of this red category of product should be increased, while the rest should be decreased.

(3) The preference of using the customization module: Each user has his/her preferences about using product customization module. The operational preferences of users are automatically recorded in the Web server. After some time of self-learning and adaptation, the system will automatically become friendly to users.

2.2 Content Matching of Individual Product Customization

In the environment of individual product customization, a framework format is used to define products. When the content matching communicates, a framework format is changed into a meta-format and the communication content matching is realized by means of counting semantic distance between meta-formats. We explain the method as follows.

It is assumed that O_a is a customization expression of an individual product, O_b is an advertisement expression of an enterprise product O$_l$. We change O_a and O_b to meta-format r$_a$ and r$_b$ as the following.

$$r_a = (x_{11}, x_{12}, \ldots, x_{1n}, f_1) \tag{1}$$

$$r_b = (x_{21}, x_{22}, \ldots, x_{2n}, f_2) \tag{2}$$

Here x_{1i} and x_{2i} are framework slot values of product O$_l$, and f$_1$ and f$_2$ are fuzzy degrees of r$_a$ and r$_b$. When carrying out matching check between individual product customization and enterprise products, the weight value of each slot is W$_1$, W$_2$, ···, W$_n$, respectively. The weight value of a meta-format is W_{n+l}, the semantic distance of r$_a$ and r$_b$ is calculated by the following formula.

$$D_p(r_a, r_b) = (\sum_{i=1}^{n} W_i \times \rho^p(x_{1i}, x_{2i}) + W_{n+l} \times | f_1 - f_2 |^p)^{\frac{1}{p}} . \tag{3}$$

$\rho(x_{1i}, x_{2i})$ is the semantic distance of x_{1i} and x$_{2i}$, p is a natural number. When p → ∞, the formula is

$$\lim_{x \to \infty} D_p(r_a, r_b) = Max\{W_i \times \rho(x_{1i}, x_{2i}) + W_{n+l} \mid f_1 - f_2 \mid\}, (1 \le i \le n). \tag{4}$$

It is obvious that if the two meta-formats have the minimum distance, they are the best matching. According to matching schemes, the module of schemes is defined as

$$d_i = \sqrt{\sum_{j=1}^{n} [W_j b_{ij}]^2} . \tag{5}$$

Each matching scheme is viewed as a row vector. There is an angle between each scheme (vector) and the customized scheme of individual product (vector). The cosine of this angle is given by

$$r_i = \frac{B_i g B^*}{\|B_i\| g \|B^*\|} = \frac{\sum_{j=1}^{n} W_j b_{ij} \, g W_j}{\sqrt{\sum_{j=1}^{n} [W_j b_{ij}]^2} \, g \sqrt{\sum_{j=1}^{n} [W_j]^2}} = \frac{\sum_{j=1}^{n} W_j b_{ij} g W_j}{\sqrt{\sum_{j=1}^{n} [W_j b_{ij}]^2}} \tag{6}$$

The projection of the matching scheme on the customization scheme of individual product is $M_i = d_i g r_i$. M$_i$ is the matching degree of each scheme and the customized scheme of an individual product. It represents the goodness or badness of every matching scheme. The higher M$_i$ is, the more the product meets the requirements of users.

2.3 Interesting Value Counting of Individual Products

Individual product customization should fulfil two conditions, (1) a user has a maximum interesting value of a product, (2) the total browser time of choosing product is no more than the limited time of the user.

Definition 1. A product s has an assigned value $v_u(s)$. This value $v_u(s)$ is called the interesting degree of product s for user u, where $v_u(s)$ is used to measure interesting degree of product s for a user u.

Definition 2. When browsing a product in browser, product s needs a duration d(s), this d(s) is called the browsing time length of the product.

Let S be a subset of products. Define $V_u(S) = \sum_{s \in S} v_u(s)$ and $D(S) = \sum_{s \in S} d(s) \le D$, where D is the limited time length of user u, $V_u(S)$ is the interesting value of whole product subset S for user u, and D(S) is the browsing time length of the whole product subset S. The interesting value *$v_u(s)$* of a product s for the user u is determined by the interesting degree $i_u(s)$ of a product s for the user u and the browsing time *d(s)* of the product s for the user u. This relationship can be expressed by the following formula.

$$v_u(s) = f(i_u(s), d(s)) \tag{7}$$

We can see that the $v_u(s)$ is increased with the increasing of $i_u(s)$ and *d(s).*

3 The Customization System of Individual Product

As an application of individual product customization, a customization system of an individual motorcycle is developed as shown in Fig.1. It takes three hierarchies of the distributed structure: browser, Web server and database server. The browser terminal includes Internet Explorer, JDBC (Java Database Connectivity), Java virtual machine and download JavaApplet from the server. The information stored in database is browsed and queried by means of Web page but the main function of the system is located in the server.

Individual motorcycle customization under the pattern of long distance and different places are completed through multi-agents. The customization system consists of user interface agents, task agents and information agents. The user interface agents interact with users by means of receiving motorcycles customization requirements from users. After forming the designing plan of an individual motorcycle and querying, and then exchanging information with other agents, the task agents are triggered. The task agents possess the domain knowledge of motorcycles customization which includes assistant agents and information agents associated with the motorcycles customization. The assistant agents have functions of resolving conflicts and syncretizing information and are responsible for information feedback of browsing and customization of motorcycles.

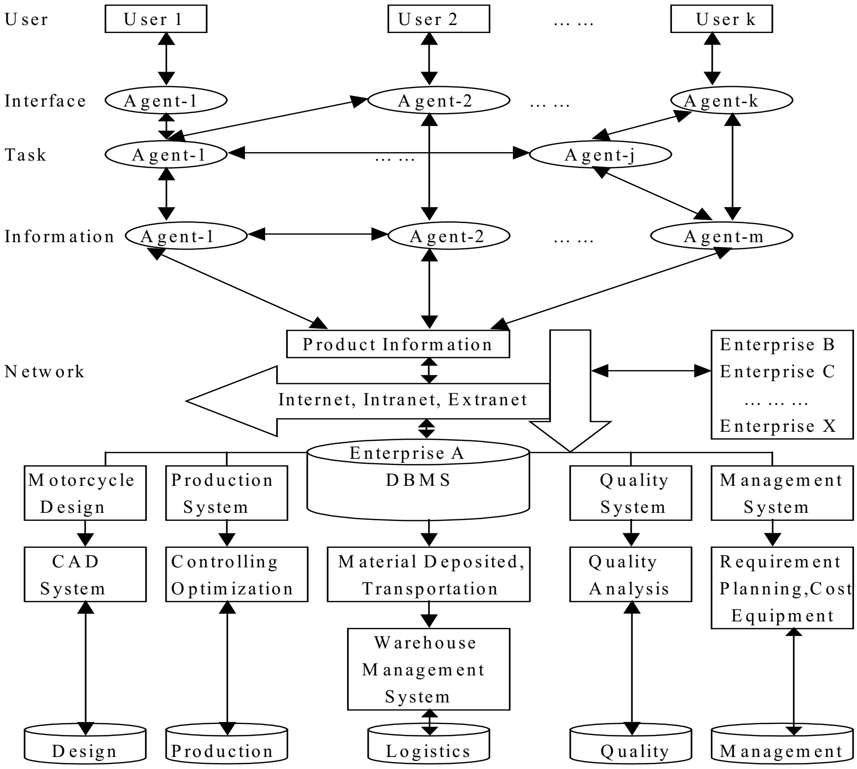


Fig.1. System Structure of Individual Motorcycles Customization

4. Case Study: Individual Motorcycle Customization

4.1 Individual Motorcycle Customization Characteristics

A motorcycle production process of manufacturing enterprises is divided into Manufacturing to Order (MTO) and Manufacturing to Store (MTS). A motorcycle production of MTO is carried out according to motorcycle functions, individual style and deliverying time required by consumers, its key to arranging production plan is controlling production speed to assure motorcycles that are produced before deliverying time. A motorcycle production of MTS is carried out according to production plan of enterprises, the key is to guarantee the safe storage quantity according to users requirements. The quota plan production made by motorcycle enterprise in every production cycle and with the lowest storage quantity is MTS.

4.2 Requirement Quantity Determination of Individual Motorcycle Customization

The requirement quantity of individual motorcycle customization includes forecasting requirement (storage production plan) and requirement in practice (customization of users). How to produce gross requirement according to the two requirements is one of the important problems. Because most motorcycle enterprises are blending types of MTO and MTS, different production pattern takes different unite method. As to MTO, gross requirement is generated according to orders of consumers for forecasting requirement is not useful. As to MTS, user order and forecasting requirement are synthesized under consideration and the bigger one between them is gross requirement.

Algorithm of gross requirement quantity

```
If plan time is in the RTL ago then        // first duration (zone 1)
    GrossRequir = OrderRequire             // Gross requirement is determined by orders.
e1se if plan time is in the PTL ago then   // second duration (zone2)
    if currTerm is MTO then                // MTO
    GrossRequire = OrderRequire            //Gross requirement is determined by orders.
    else                                   // MTS
    GrossRequire = Max(OrderRequire， Forecast)
    Endif
else                                       // third duration (zone3)
    GrossRequire = Forecast                // long-term plan
End if
```

In this algorithm，GrossRequire is a gross requirement, OrderRequire is a customization requirement, Forecast is a forecasting requirement, RTL is the requirement time limit, PTL is the planning time limit, as shown in Fig. 2.

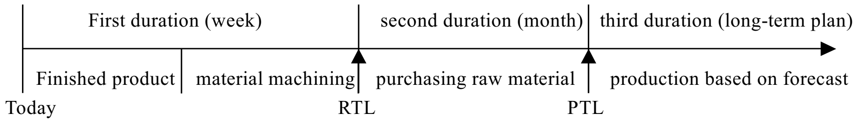


Fig.2. Requirement Quantity Determination of Individual Motorcycle Customization

4.3 A Process of Individual Motorcycle Customization Based on Multi-agent Technology

Traditional working pattern of information searching and motorcycle customization is shown in Fig. 3(a): two communication sides exchange information through address route. The used catalog agent classifies information resource to reduce searching scope basically and may not have formalizing knowledge of information service. So it is very hard to bring into play the communication agent function with agility and effectiveness in the large network environment.

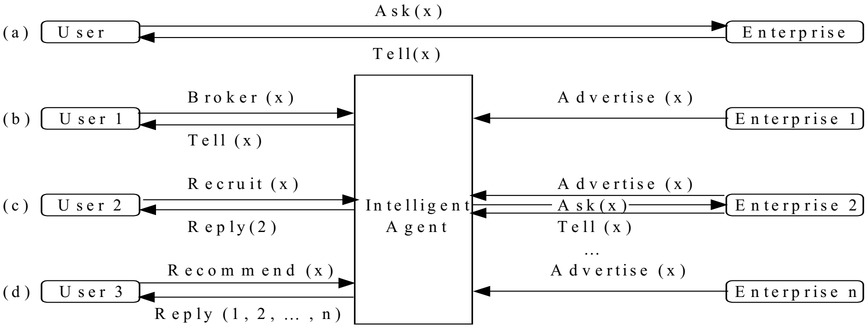


Fig.3. Several Patterns of Information Route

Information service based on intelligent agent builds up a bridge between users and enterprises. Enterprises send motorcycle information to intelligent agents by sentences in the communication pattern, for example, here is a product sentence:

(Advertise (Content (ask_about (Content (motorcycle? x)))))

It expresses information requirement for “(motorcycle? x)” .

Concerning on the foundation of three kinds of message (broker, recruit, recommend) provided by KQML, an intelligent agent is designed. The agent provides three searching methods of information as shown in Fig.3 (b), (c), (d). If the intelligent agent receives n information sentences of motorcycle x, the working process under three patterns is as follows.

- If users send sentence broker (x) to the intelligent agent, a suitable enterprise is chosen by the intelligent agent, and it sends a querying sentence. When receiving an answer from an enterprise, the intelligent agent sends back the answer to user as shown in Fig. 3(b).
- If users send sentence recruit (x) to the intelligent agent, a suitable enterprise is chosen by the intelligent agent and the enterprise address is fed back to users. Then users link the enterprise directly with the address as shown in Fig. 3(c).
- If users send sentence recommend (x) to the intelligent agent, all information of enterprises providing motorcycle x is shown to users by the intelligent agent and the enterprise is chosen to link with by users as shown in Fig. 3(d).

4.4 An Individual Motorcycle Customization Calculation

We take the scheme customization of an individual motorcycle as an example to illustrate product customization matching degree calculation. There are three types of motorcycle schemes, two of them are former scheme 1 and former scheme 2, the third one is a new designed motorcycle scheme. Now the information associated with the three motorcycle schemes is processed to determine whether or not one of the three schemes is satisfying the users requirements of an individual motorcycle. If one of three schemes satisfies the requirements of users’ individual motorcycle, then the scheme is chosen to make the individual motorcycle, otherwise, the enterprise should design the fourth or the fifth motorcycle scheme and determine whether or not it meets the requirements of users' individual motorcycle again until a satisfied motorcycle scheme is chosen. The indexes of individual motorcycle customization are shown in Table 1.

Table 1. Indexes of Individual Motorcycle Customization

Index	New Designed Scheme	Former Scheme 1	Former Scheme 2
Price (RMB yuan)	2200	1100	600
Weight (Kilogram)	79	79	90
Friendlyness	0.90	0.72	0.65
Reliability	0.85	0.78	0.70
Interest Index I1	1	0	0
Interest Index I2	2	1	0
Interest Index I3	1	0	0
Interest Index I4	1	0	0
Interest Index I5	4	1.5	1
Total of Interest Index	9	2.5	1
Browse Time T1	4	2	1
Browse Time T2	2	1	1
Browse Time T3	1	0	0
Browse Time T4	1	0	0
Browse Time T5	2	1	0
Total of Browse Time	10	4	2

All of above data are input into the matching calculation module of an individual motorcycle customization system.

1 5 3 # 1 is the number of cost index, 5 is the number of benefit index, 3 is the number of decision scheme. #

The matrix of indexes is as followings.

```
2200  79  0.90  0.85  9  10
1100  79  0.72  0.78  2.5  4
 600  90  0.65  0.70  1   2
0.40 0.10 0.15  0.20  0.25  0.40  # Weight #
```

The output result of module is the followings. The input matrix:

```
2200.000000 79.000000 0.900000 0.850000 9.000000 10.000000
1100.000000 79.000000 0.720000 0.780000 2.500000 4.000000
600.000000 90.000000 0.650000 0.700000 1.000000 2.000000
0.400000 0.100000 0.150000 0.200000 0.250000 0.400000# Weight#
```

Middle Calculation Matrix:

```
0.000000 0.000000 0.222375 0.296500 0.370625 0.592999
0.381214 0.000000 0.062265 0.158133 0.069492 0.148250
0.592999 0.148250 0.000000 0.000000 0.000000 0.000000
0.592999 0.148250 0.222375 0.296500 0.370625 0.592999
```

The results of evaluation indexes:

```
d0=0.791438   r0=0.791438      M0=0.626374
d1=0.448346   r1=0.893193      M1=0.400460
 d2=0.611250    r2=0.611250    M2=0.373626
```

Here d0, d1, d2 are the module of every matching scheme respectively; r0, r1, r2 are cosine of angle between every matching scheme and users customization scheme; M0, M1, M2 are projections of every matching scheme on the customization scheme of user individual motorcycles. According to the results compared among M0, M1, M2, the new scheme of the individual motorcycle M0=0.626374 meets more users' requirements than the other two, so the new scheme of the individual motorcycle should be the first choice to produce for users.

5 Conclusions

The principle of consumer individual product customization among the remote locations is studied and the calculating formula of interest value of product is put forward.

The system of individual motorcycle customization is developed. This system is composed of consumer interface agents, task agents and information agents and taken three hierarchies distributing structure of browser, web server and database server.

The application practice of motorcycle enterprise shows that the individual customization of enterprise product is an effective measure to enhance market competition ability of enterprise products. The successful application of the system overcomes the limitation and shortage of traditional product development and market currency from roles of enterprises and users.

References

Finin, T., Weber, J. (1993), "Specification of the KQML: agent-communication language," *DARPA Knowledge sharing initiative*, External interfaces working group.

Finin, T., et a1.(1994), "KQML as an Agent Communication Language," CIKW'94.

Huhns, M. N., Stephens, L. M. (1999), "Multiagent systems and societies of agents," In G. Weiss, *multiagent systems: A modern approach to distributed artificial intelligence*, Cambridge, MA: MIT press.

O'Leary, D.E.D., Plant, R. (1997), "Artificial intelligence and virtual organizations," *Communications of the ACM*, Vol.40, pp.52-59.

Sycara, K., Pannu, A., Williamson, M., Zeng, D. (1996), "Distributed intelligent agents," *IEEE Expert*, pp.36-45.

Wüpping. J. (2000), "Integrated Approach to Get Good Customer Product-solutions," *Engineering Design*, Vol.1, pp.49-52.

Zwass, V. (1991), "Electronic commerce: Structures and issues," *International Journal of Electronic Commerce*, Vol.1, pp.3-23.

An Intelligent Design Method of Product Scheme Innovation*

Yong Zhao[1], Cheng Zhao[2], Jianzhong Cha[1]

[1]School of Mechanical and Electronical Control Engineering, Beijing Jiaotong University, Beijing, 100044, China

[2]Department of Mathematics and Computer Science, Indiana State University, Terre Haute, IN 47809, U.S.A.

Abstract: The main objective of this intelligent designing research is to provide an astute environment for designers. This type of environment is to be designed and processed so that it is compatible with a person's thoughts. In this environment the designers' thoughts are effectively taken to process functional requirements of products. The concept of designing and evaluating the schemes of the products can be achieved and furthermore optimized, the structure designing and its manufacturability analysis are carried out. This paper studies cell classes of the functions and structures. By the use of artificial intelligent technology the knowledge required and the innovative scheme expressed are developed. As an example, an intelligent design of a motorcycle is used to illustrate the application of the method to produce an innovative scheme.

Keywords: product scheme innovation, intelligent design method, motorcycle

1 Introduction

Intelligent design of product scheme innovation is a creation requirement of enterprise product for market economic. It is a synthesis process of analyzing market requirement, developing product conception and defining product for an enterprise and has characteristics of: (1) complex data model which concerns with the information of load, precision, space geometry, mechanics, figures and materials, (2) multi-functions, (3) easy changing requirements of product for users. So small batch, multi-type designs of product are popular in enterprise production and it is required an effective design method of product particularly scheme design.

* This work was funded by the NSFC under the grant 50335040.

A product scheme design is a creative activity of non-numerical value calculation by means of thinking, reasoning and judging. Creative activities often produce product innovations from basic principles. The resulting products will have various types, more useful and effective functions, and lower prices. So the core of a product scheme design is an innovation design. Usually, it is based on new principles and new methods from traditional design experiences and new technology.

In this research field, one of the typical designs is a structure design method of function cell. It was studied by MerviRanta (1996). The advantage in this method is how to map function descriptions into parts and assembly shapes, sizes and relationships among them. The key changing process is the map process, that is a three-hierarchy method of function, structure and product scheme. Alan, B. (1994) and Runkel, J. T. (1996) published a study on the model of general design theory (GDT). They thought that the key to design is a decomposition process, a map process and a synthetic process. They listed all the basic function cells and realization methods according to function catalog. No matter what changes of design scheme, their basic cells were gained from the function catalog. The function decomposition process is a hierarchy decomposition of design requirements and it ends when sub-function is realized by basic function cell. The map process is choosing parts that satisfy design requirements from function catalogs according to the characteristics of basic functions. The input elements are the function requirement descriptions and some constraint conditions. The output elements are parts and assemblies that satisfy design requirements. The comprehensive process is combining parts and assemblies into products according to the relationship among every basic function cell provided by decomposition process. The reliability of this method greatly depends on the function decomposition way and its function catalog. But the design result is not controlled effectively because it is an open system. Tetsu (1994) describes innovation design of product scheme from requirement description. The requirements are described into functions, behaviors and attributes. These three kinds of descriptions are gradually detailed, corrected and tested until a product is produced. The whole design is a cycle process of sub-cell designing. The advantage of this method is: it has valuation hierarchy of the result, feedback control is permitted to the result, and the product is optimized. So it is a closed loop design method according to the thinking custom of product designer. But the cognition model of scheme design is not suitable for the practice design process of product.

For concurrent intelligent design and assembly planning, X.F. Zha (2002) proposes a novel knowledge intensive multi-agent cooperative/collaborative framework which integrates product design, design for assembly, assembly planning, assembly system design, and assembly simulation. An AI protocol based method is proposed to facilitate the integration of intelligent agents for assembly design, planning, evaluation and simulation process. A unified class of knowledge intensive Petri nets is defined using the object oriented knowledge-based Petri net approach and used as an AI protocol for handling both the integration and the negotiation problems among multi-agents. The detailed cooperative/collaborative mechanism and algorithms are given based on the knowledge objects cooperation formalisms. As such, the assembly-oriented design system can easily be implemented under the multi-agent-based knowledge-intensive Petri net framework with concurrent integration of multiple cooperative knowledge sources and software. Thus, product design and assembly planning can be carried out simultaneously and intelligently in an entirely computer-aided concurrent design and assembly planning system. A. Gayretli, H.S. Abdalla (1999) discusses an object-oriented constraints-based system for concurrent product development. This approach enables designers to evaluate and optimize feasible manufacturing processes in a consistent manner as early as possible during the design process. This helps in avoiding unexpected design iterations that wastage a great amount of time and effort, leading to longer lead-time. The development process has passed through the five major stages: Firstly, an intelligent constraint-based design system for concurrent product and process design has been developed. Secondly, a manufacturing process optimization module has been constructed. Thirdly, the product features, processes, cost, time and constraints to be used for carrying out various design tasks has been represented in the format of constraints, frames, objects, and rules. Fourthly, the process optimization and evaluation rules for the selection of feasible processes for complex features, and finally, the information management system that ensures consistency in information exchange and decision making activities have been developed.

As conceptual process design is a multi-agent, distributed problem-solving activity that can be significantly enhanced with the aid of several individual intelligent systems. It is quite difficult to develop such intelligent systems on the basis of existing expert system shells because of the complexity of conceptual process design. Q. Wang, et al (1995) analyzed the characteristics and requirements of conceptual process design and an intelligent design environment (Meta-COOP) for conceptual process design is presented. This environment provides such distinct features as the integration of various knowledge-representation and inference methods, and deals with multimedia information.

Product designs can fail for technological reasons and also for human reasons. Both computer-aided design (CAD) tools and formal methods have attempted to limit the risk of product failure by logical assessment of the design and its technology. However, to improve design effectiveness and efficiency the contribution of people to a design’s risk of failure must also be assessed and controlled. A method is presented of controlling and limiting human design through the application of a four-path model of design developed by P.F.Culverhouse (1995). The model categorizes designs as repeat design, variant design, innovative design or strategic design. Each has its own set of pre-existent constraints that limit the capacity for unwanted innovation to the product at the outset of each development program. This provides a mechanism by which designer creativity may be controlled. The manner in which designers are constrained may also be applied to the control of CAD tools commonly employed in the design process.

Although many models and model-based design methods have been proposed, in practice their effectiveness has tended to be limited to solving fragmented design problems. One factor that has diminished the effectiveness of model-based interaction design approaches has been the inability to integrate the various models used for different aspects of the overall design problem. Dong-Seok Lee, Wan Chul Yoon (2004) proposes a novel approach for combining a structural model and a functional model for complicated interaction design. Formal correspondence between the models is defined and a conversion process to transform from one model to the other and vice versa is introduced. The functional model, OCD, is an efficient technique for representing task procedures, while the structural model, state chart, is well suited to representing system behavior. The usability needs and system requirements are introduced into the design process through either representation. Then, the constraints formed by a decision in a model can be seen by the designer in the other model through transformation. The possibility of automatic conversion between the models warrants the consistence between the models through the design process even when the models should continually evolve. As design usually involves the integration of diverse, sometimes conflicting, concepts and requirements into a coherent single composition. John H., Paul K., Edith A. and William P. (2000) proposes a method for negotiating design across domains, by examining issues of perception, generation and evaluation, and detailing a prototype in which these mechanisms are augmented using computational agents for achieving coherence and innovation in remote collaborative design. Filter Mediated Design is intended to explore the processes and strategies of carrying out intelligent designs and design intelligence.

In this paper an intelligent design method of product scheme innovation by means of artificial intelligent technology is put forward. The research status and characteristics of intelligent design of product scheme innovation are described in section one. In section two, the principle and model of intelligent design of product scheme innovation is proposed and studied, the function cell class, structure cell class and acquiring, expressing, reasoning of product scheme design knowledge are studied in detail, the intelligent design method is utilized in scheme design of motorcycle and concrete design steps are presented. And the research conclusion is given in section three.

2 Intelligent Design Principle of Product Scheme Innovation

2.1 Intelligent Design Model of Product Scheme Innovation

Intelligent design cells of product innovation are the abstract classes that have relatively independent abilities. They are used to express special functions and host some geometry characteristics in the process of product scheme. These kinds of design cells consist of some function structures. Cell granularity might be a part or some shape cells in a part or an assembly. The attributes of cells include function attributes and geometry attributes. Function attributes define the special function requirements of product composing. Geometric attributes express and accomplish the important geometry characteristics of functions and scheme relationships between product synthesis hierarchies.

Association among cells is given by structure, link and message. There are two kinds of cell class structures. One is assembling structure and the other is assorting structure. The assembling structure is made up of object classes of the product. A product is composed of some moving mechanisms of realizing function requirements and structure schemes as supporting framework. Every moving mechanism or structure is composed of sub-assemblies and parts or parts and sub-structures. And at last the assembling structure is formed by object classes. The assorting structure consists of basic classes and the derived classes are defined on foundation of basic classes. The basic classes include designed parts and parts from markets. Apart from the strong association presented by structures among scheme cell classes of product, the link styles and message mechanisms are also used to form an integrated innovation scheme in feebleness association. So the model representation of product scheme designing is as following.

[Product scheme] = [Function cell classes of a product] × [Structure cell classes of a product] × [Knowledge reasoning engine of the product designing]. (1)

2.2 Function Cell Classes of a Product

In the designing process, the function decomposition is an important step. This step a simple sub-function or several function cells induced by structures. The description of function includes the function meaning and the function link relationship. The function meaning is the assorting description of function types for a product scheme. It avoids semantic coincidence between function words. The function link relationship represents a hierarchy relationship and a realization method among function descriptions which include a join relation of the function decomposition, an extract relation of the function selection, a time relation of the function existence, a consequence relation of the function composition, a message relation of function existence, a renovating relation of function creation, changing and vanishing. So the description of function cell classes of a product given by:

FDL=(FN, FB, FO, (FP, FN)) (2)

where FN is a function name, FB is the main function behavior, FO is a function object, (FP，FN) is up and down function relations in functional hierarchy.

2.3 Structure Cell Classes of a Product

A function can be obtained through a function cell in a part or some action among different parts. In order to make parts and assemblies form a product on foundation of scheme cells of parts and assemblies, map rules both from a function to a scheme cell and from scheme cell to parts are needed to set up.

As detailed geometry description is not involved in a product scheme design, the main geometry parameters in a structure cell scheme are represented by vectors. The places where the scheme cells in the product scheme structure and the local coordinate origin of scheme cells are determined by starting point of vectors. The assembling direction in a product scheme model is represented by a vector direction. Other parameters of the structure cell scheme are defined by the conception of the compound vector module length. The compound vector module is a multi-dimension vector with recording structure parameter information. The geometry encircling box, the main sizes of structure and function characteristic requirements such as surface coarseness degree, matching precision and so on are expressed by sub-vectors. So the vector of structure cell scheme is given by:

$$VF\{(x_0,y_0,z_0), (\alpha, \beta, \gamma), (box, md, pre)\} \quad (3)$$

where (x_0, y_0, z_0) represents local coordinate origin of structure cells, (α, β, γ) represents angles between structure cell and X, Y, Z coordinate axes, $0 \leqslant \alpha, \beta, \gamma \leqslant 180°$, box means geometry surrounding box of structure cell in compound model, md means the main dimensions defined by structure cell, pre means the main matching precision of a structure cell.

The structure cell in product scheme model is represented as follows.

$$F=(FN, FID, FFL, GPT, RPT, VF) \quad (4)$$

where FN and FID are the name and mark of structure cell scheme, FFL is the surface link table in composing of scheme cell, GPT is a surface pointer of scheme cell, RPT represents a relation pointer among product scheme cells, VF is a vector of structure cell.

2.4 Knowledge Acquiring, Expression and Reasoning of Product Scheme Intelligent Design

The concrete process of the link between a function and a structure scheme cell is an intelligent innovation process of scheme cells. First, the structure cells associated with functions are selected as intelligent innovation cells in a basic scheme. The selection in a basic scheme has two rules. One is selecting important functions of a product and the other is selecting cells that have more relations with scheme cells or parts. Second, scheme cells associated with functions in parts and assemblies are selected to synthesis and compare with basic scheme cells in order to gain a logical, effective new cell scheme of parts and assemblies.

The relation between a function and a structure is expressed as $RF=(F_i, i, E)$, where F_i is a function set, i is a structure set, E is a relation between function and structure that is determined based on knowledge.

Design knowledge is acquired from domain engineers or design specialists. The main resources of knowledge are design specialists and special technology information. The tool for acquiring knowledge, such as knowledge editor, is utilized to transform knowledge into inner expression recognized and stored by computer in order to examine knowledge, check syntax and so on.

Because of characteristics of a multi-process and a multi-object in a scheme intelligent design, the designing objects are decomposed into a series of sub-objects with some independence. All relations among sub-objects form public knowledge base. The designing knowledge of every sub-object forms an independent sub-base of knowledge. The knowledge in public base is inherited. The hierarchy knowledge model controlled by the meta-knowledge and then directed by a hierarchy frame is taken. In this model, the knowledge expression methods of a rule and a frame are combined with each other. The scheme design data are put into the qualification part of the rule knowledge. Concrete design methods, algorithms and so on are put into the conclusion part and the reliability method is taken to express fuzzy knowledge as shown in Fig. 1.

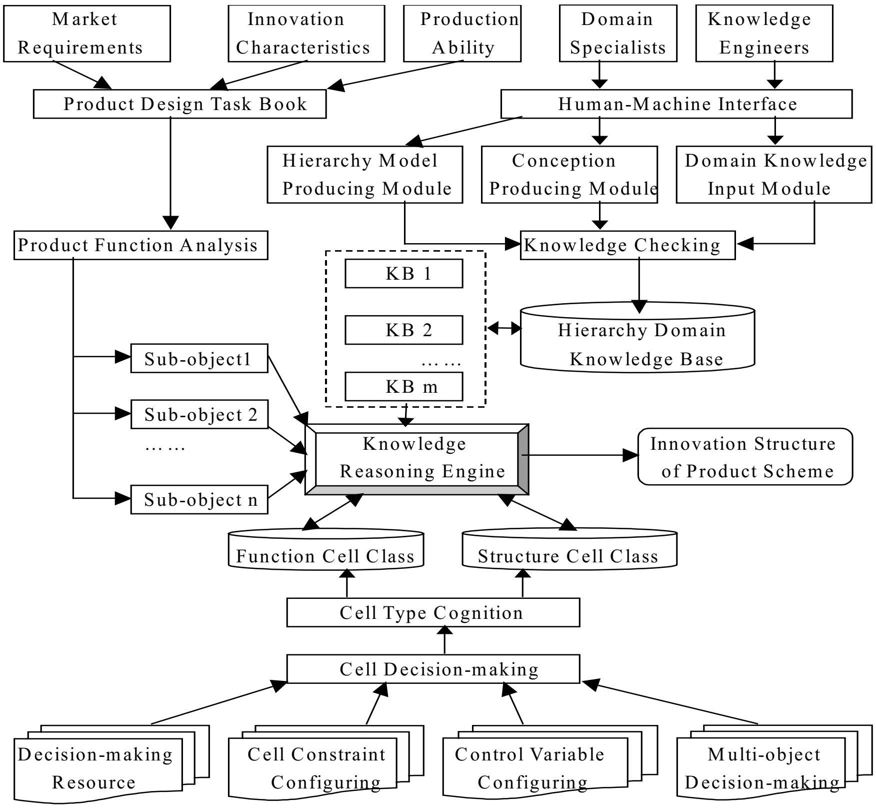

Market Requirements
Innovation Characteristics
Production Ability
Domain Specialists
Knowledge Engineers
Product Design Task Book
Human-Machine Interface
Hierarchy Model Producing Module
Conception Producing Module
Domain Knowledge Input Module
Product Function Analysis
Knowledge Checking
KB 1
KB 2
……
KB m
Hierarchy Domain Knowledge Base
Sub-object1
Sub-object 2
……
Sub-object n
Knowledge Reasoning Engine
Innovation Structure of Product Scheme
Function Cell Class
Structure Cell Class
Cell Type Cognition
Cell Decision-making
Decision-making Resource
Cell Constraint Configuring
Control Variable Configuring
Multi-object Decision-making

Fig. 1 Knowledge Acquiring and Hierarchy Expression Model

2.5 Intelligent Design of Motorcycle Innovation

Motorcycle is one of the important vehicles in China. Each driver's safety depends on its quality and reliability. The key to motorcycle designing is its scheme design. Starting from requirements of a motorcycle, designing problems are exactly described from functions and behaviors in early stage of designing, designing qualifications and quality requirements are put forward, at last the designing task is written as a paper contract. Intelligent design of a motorcycle innovation is formed based on experience, theoretic knowledge on existing scheme and also based on economic analysis. Fig. 2 is intelligent design process of motorcycle innovation.

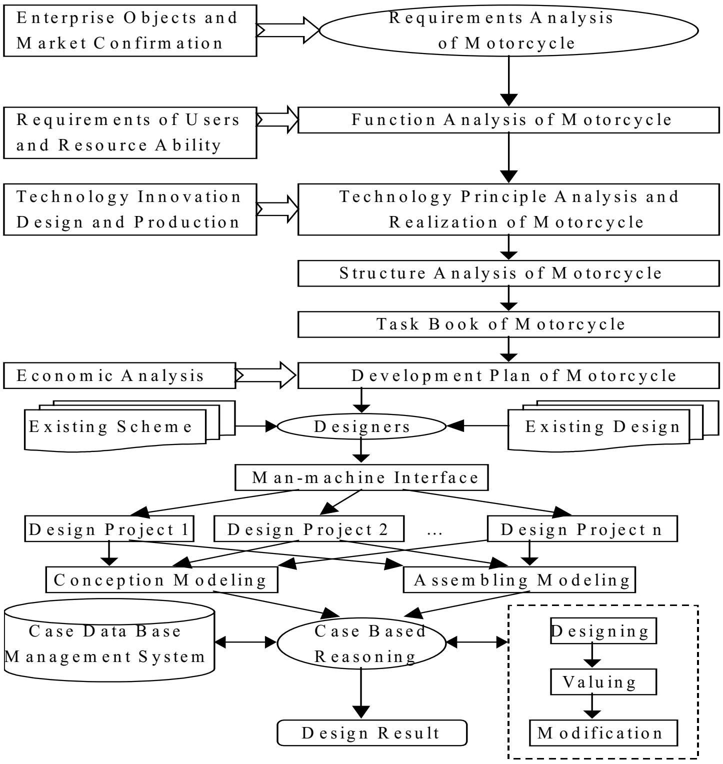

Enterprise Objects and Market Confirmation
Requirements Analysis of Motorcycle
Requirements of Users and Resource Ability
Function Analysis of Motorcycle
Technology Innovation Design and Production
Technology Principle Analysis and Realization of Motorcycle
Structure Analysis of Motorcycle
Task Book of Motorcycle
Economic Analysis
Development Plan of Motorcycle
Existing Scheme
Designers
Existing Design
Man-machine Interface
Design Project 1
Design Project 2
...
Design Project n
Conception Modeling
Assembling Modeling
Case Data Base Management System
Case Based Reasoning
Designing
Valuing
Modification
Design Result

Fig. 2 Intelligent design process of Motorcycle innovation

The designing of motorcycle has many knowledge types and complex knowledge structures including handbook knowledge, chart knowledge, formula knowledge, decision-making knowledge and expert experience knowledge. Designer knowledge in brain is not disorder and unsystematic and in fact is an organic integer composed of knowledge modules linked in different hierarchy. According to the characteristics of design knowledge, semantic network model based on rules is taken to illustrate describing, organizing, acquiring and managing the knowledge of scheme innovation design.

The instance reasoning technology is taken to design the link between function and structure of a motorcycle from three sides of design, estimating and modification. The rule reasoning types of orientation, direction, join and movement are taken into account in the process of designing. The structure cell is decomposed according to constitution orientation characteristics of a motorcycle. The network model of a complex case is changed into a tree structure consisting of a chain table and many two-branch sub-trees. The tree structure utilized in practice is represented by knowledge rules. The motorcycle designed at last is expressed by root node in the tree structure. The complex structure cell is expressed by non-leaf node and the basic structure cell is a leaf node. One leaf node with many fathers in the model implies that the same structure cell can be used many times in the same case scheme. It is prescribed that the left children of non-leaf node are the innovation cell of basic scheme. Operation relationships of merging, difference and adjoining between right children and left children are expressed by structure cell vectors that are (1) position relationship (dx, dy, dz), (2) direction relationship (d α ,d β ,d γ), (3) compounding module length relationship. The relationship of compounding module length is a constrained parameter relationship between two structure innovation cells such as scheme size relationships including more than, less than and equal to relationships. As approximate position relationship between structure cells is dealt with scheme innovation designing, most relationships of constrained parameter are more than or less than relationship.

The following are the design steps of motorcycle innovation.

Step 1. Designing task paper work for a motorcycle is changed into equivalent requirement sets of function (because every function requirement is needed to map into one or many scheme cells.)

Step 2. Function requirements are divided into satisfied sets and unsatisfied sets, all function requirements are put into unsatisfied sets. When unsatisfied function sets are empty, it is shown that the inputs are invalid. Exit and making satisfied sets empty and scheme structure model is empty.

Step 3. If an unsatisfied set is empty, go to Step 6, else choose a requirement.

Step 4. Function requirements are mapped into scheme cells, choose a cell as a basic scheme cell. The following processes are carried out for every scheme cell.

Function decomposition. The described function requirements are decomposed, merged and deleted according to design rules and knowledge.

Structure synthesis. Function requirements are mapped into scheme cells based on knowledge reasoning. If the structure cell mapped is a non-leaf node cell of the shape scheme, further decomposition is made and a new requirement of a function is formed. Go to Step 3.

Cell operation of scheme innovation. Scheme cells are put into the structure model of a motorcycle with a suitable cell operation.

Scheme evaluating. The analyses of function and geometry constraints of a motorcycle structure are made. If not satisfied, go to (1).

Step 5. The examined and satisfied function requirements are deleted from unsatisfied sets. Put them into satisfied sets. Go to Step 3.

Step 6. The structure cells are translated into shape features according to the innovation model of scheme structure and all attributes of a motorcycle are further determined.

3 Conclusions

In order to provide an implement mechanism of intelligent innovation design and fulfill the unification expression of design information, principle designing of motorcycle is combined with intelligent design technology of cells innovation, corresponding relationship between function model and structure model of motorcycle is also proposed. Further study can be done in two hierarchies. One is calculation model hierarchy. The intelligent design of product innovation is needed to carry out from two sides of a function and a structure based on the foundation of a unifying product data model. The other is artificial intelligent technology hierarchy. The thinking process of the human innovation design will be studied from recognizing behaviors and knowledge based reasoning model of the innovation design.

References

Alan, B. (1994), "DIDS: Rapidly Prototyping Configuration Design Systems," *Journal of Intelligent Manufacturing*, NO.5, pp.33-45.

Culverhouse P. F. (1995), "Constraining designers and their CAD tools," *Design Studies,* 16, pp.81-101.

Dong-Seok L., Wan-Chul Y. (2004), "Coupling structural and functional models for interaction design," *Interacting with Computers*, 16, pp.133–161.

Gayretli A., Abdalla H. S. (1999), "An object-oriented constraints-based system for concurrent product development," *Robotics and Computer-Integrated Manufacturing*, 15, pp.133-144.

John H., Paul K., Edith A. and William P. (2000), " Filter mediated design: generating coherence in collaborative design" *Design Studies,* **21, pp.**205–220.

Mervi, R. (1996), "Integration of Functional and Feature-based Product Modeling-the IMS/GNOSIS Experience," *CAD J.*， Vol.**28**, NO.5, pp.371-381.

Runkel, J. T., (1996), "Domain Independent Design System Environment for Rapid Prototyping of Configuration Design System," In: *Proceeding of the 2nd International Conference on AI in Design AID' 92*, pp.21-40.

Tetsu. (1994), "From General Design Theory to Knowledge-intensive Engineering," *Artificial Intelligent for Engineering Design Analysis and Manufacturing*， NO.8, pp.319-333.

Wang Q., Zhu J. Y., Shu Y. Q., Rao M., Chuang K. T. (1995), "An Intelligent Design Environment for Conceptual Process Design, " *Engineering Application of Artificial Intelligence,* 8,No.2,pp.115-127.

Zha, X. F. (2002), "A knowledge intensive multi-agent framework for cooperative/collaborative design modeling and decision support of assemblies," *Knowledge-Based Systems*, 15, pp.493–506.

Communication Method for Chaotic Encryption in Remote Monitoring Systems for Product e-Manufacturing and e-Maintenance

Chengliang Liu[1], Kun Xie[1], Y.B. Miao[1], Xuan F. Zha[1,2], Zhengjin Feng[1], Jay Lee[3]

1-Institute of Mechatronics Control, Shanghai Jiao Tong University Shanghai 200030, China; 2-National Institute of Standards and Technology, Gaithersburg, MD 20899, U.S.A; 3-Center for Intelligent Maintenance, University of Wisconsin at Milwaukee, WI 53211, U.S.A

Abstract. In chaotic cryptosystems, it is recognized that using (very) high dimensional chaotic attractors for encrypting a given message may improve the privacy of chaotic encoding. In this paper, we study a kind of hyperchaotic systems using some classical methods. The results show that besides the high dimension, the sub-Nyquist sampling interval is also an important factor that can improve the security of the chaotic cryptosystems. We use the method of time series analysis to verify the result.

Keywords: remote monitoring; time series analysis; surrogate data; e-manufacturing; e-maintenance

1. Introduction

For the last decade synchronization of chaotic systems has been explored very intensively by many researchers in various fields ranging from physics, mathematics to engineering for possible applications in communication systems (Parlitz and Kocarev 1992, 1996, 1997). Typical examples include engineering systems for e-maintenance and e-manufacturing where mass data/information transmitted on-line or real-time is regarded (Koc and Lee 2001). Parlitz and Kocarev (1997) used the surrogate data analysis to unmask chaotic communication systems. Their research result shows that one possibility to improve the privacy of chaotic encoding is to use (very) high dimensional chaotic attractors for encrypting a given message (Parlitz 1998).

First, based on the active-passive decomposition method, a hyperchaotic system is decomposed into the transmitter and receiver.

$$\begin{aligned} \dot{x}_1 &= -x_2 + ax_1 \\ \dot{x}_m &= x_{m-1} - x_{m+1} \\ \dot{x}_M &= \varepsilon + bx_M\left(x_{M-1} - d\right) \quad 1<\mathrm{m}<\mathrm{M} \end{aligned}$$

where, m is the embedded dimension, M is the total dimension, $a = 0.29$, $b = 4$, $d = 2$, and $\varepsilon = 0.1$. For $M = 11$, its chaotic attractor has $D_L = 10.02$ Lyapunov dimension; for $M = 101$, the dimension is $D_L = 100.02$. The transmitter and receiver are expressed as follows:

transmitter:

$$\begin{aligned} \dot{x}_1 &= -x_2 + (a-1)x_1 + s \\ \dot{x}_m &= x_{m-1} - x_{m+1} \qquad (m = 2, \cdots, M-1) \\ \dot{x}_M &= \varepsilon + bx_M\left(x_{M-1} - d\right) \end{aligned} \tag{2}$$

transmitted signal: s=x_1+i, $i(t)$=0.2$sin(t)$;

receiver:

$$\begin{aligned} \dot{y}_1 &= -y_2 + (a-1)y_1 + s \\ \dot{y}_m &= y_{m-1} - y_{m+1} \qquad (m = 2, \cdots, M-1) \\ \dot{y}_M &= \varepsilon + by_M\left(y_{M-1} - d\right) \end{aligned} \tag{3}$$

With a sampling interval t_s=1.4, four cases are compared and simulated: (a) i=0, M=1; (b) i=0.2$sin(t)$, M=11; (c) i=0, M=101; and (d) i=0.2$sin(t)$, M=101. In these four cases, the phase graphs possess no discernible structure and thus may not be easily distinguished from a linear stochastic process.

Second, to investigate the efficiency of the hyperchaotic system in masking the sinusoidal information, two surrogate data tests are applied for deterministic nonlinearties to the transmitted signals. The analysis showed that the system described in Eq. (1) exhibits nonlinear noise-like dissipative dynamics. It was therefore necessary to use (very) high dimensional chaotic carriers to achieve a satisfactory degree of privacy.

However, their method only used surrogate data to validate the security of the hyperchaotic system. This kind of system is not absolutely safe. It cannot stand up other test methods. Through our research, we find that when using the VWK (Volterra- Wiener- Korengerg test) method to verify it there will be not the same security as the surrogate data method. In this paper, we study why this hyperchaotic system is vulnerable in the VWK test and propose a solution to improving its security. We use both the VWK method and the surrogate data method to make comparison.

2. Nonlinear Test Based on the VWK Method

2.1 The Theory

For a dynamic system, let input and output sampling points be $\{x_n\}_{n=1}^N$, $\{y_n\}_{n=1}^N$, SI (sampling interval) be τ , and the length of the data be N . If $x_n, x_{n-1}, \cdots, x_{n-k+1}$ are used, the discrete Volterra series can be expanded by the Taylor polynomial of y_n . k is the order of the system. Barahona (1996) presented a kind of the closed loop Volterra series using y_n feedback ($x_n = y_{n-1}$), which can be calculated through the following formula:

$$\begin{aligned} y_n^{calc} &= a_0 + a_1 y_{n-1} + a_2 y_{n-2} + \cdots + a_k y_{n-k} + a_{k+1} y_{n-1}^2 \\ &\quad + a_{k+2} y_{n-1} y_{n-2} + \cdots + a_{M-1} y_{n-k}^d \\ &= \sum_{m=0}^{M} a_m z_m(n) \end{aligned} \tag{4}$$

where $\{z_m(n)\}$ is an all-different combination composed of embedding space coordinates $(y_{n-1}, y_{n-2}, \cdots, y_{n-k})$, d is the highest combination degree, k is the model order. Since the total dimension is $M = (k+d)!/(d!k!)$, k is equal to the embedding dimension, and d is equal to the nonlinear degree of the model. By using one step forecasting error, the short time forecast power could be calculated:

$$\varepsilon(k,d)^2 \equiv \frac{\sum_{n=1}^{N} (y_a^{calc}(k,d) - y_n)^2}{\sum_{n=1}^{N} (y_n - \bar{y})^2} \tag{5}$$

where, $\bar{y} = \frac{1}{N} \sum_{n=1}^{N} y_n$, $\varepsilon(k,d)^2$ is the regularization variance of the residual.

From formula (4) and (5), we know that an appropriate *k* and *d* must be given before using this method. And the most appropriate k_{opt} and d_{opt} are just *k* and *d,* which can make the information criterion *C(r)* be the least.

$$C(r) = \log_{\varepsilon}(r) + r/N \tag{6}$$

where, N is the number of data, and *r* is the order of model. If *d*=1, VWK is the linear model; if d>1, VWK is the nonlinear model. It is noted that the larger k_{opt} is, the larger *M* is. If *d* > 1, the count quantum will be enhanced. We should adjust *k* and *d* to make $C^{nl}(r)$ be less than $C^{lin}(r)$, k=k_{opt} and d=d_{opt}.

2.2 The Effect of Sampling Interval

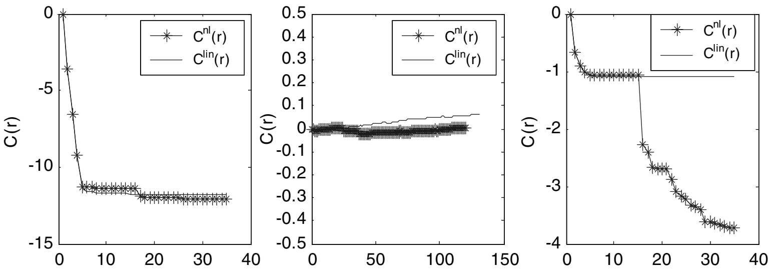
Here, we take the Lorenz chaotic system as an example to explain the relationship between the sampling interval and the VWK test. The Lorenz chaotic system is:

$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = \gamma x - y - xz \\ \dot{z} = -bz + xy \end{cases} \tag{7}$$

where, $\sigma = 10$， $\gamma = 28$， $b = 8/3$, and the chaotic signal $x(t)$ is the encryption key.

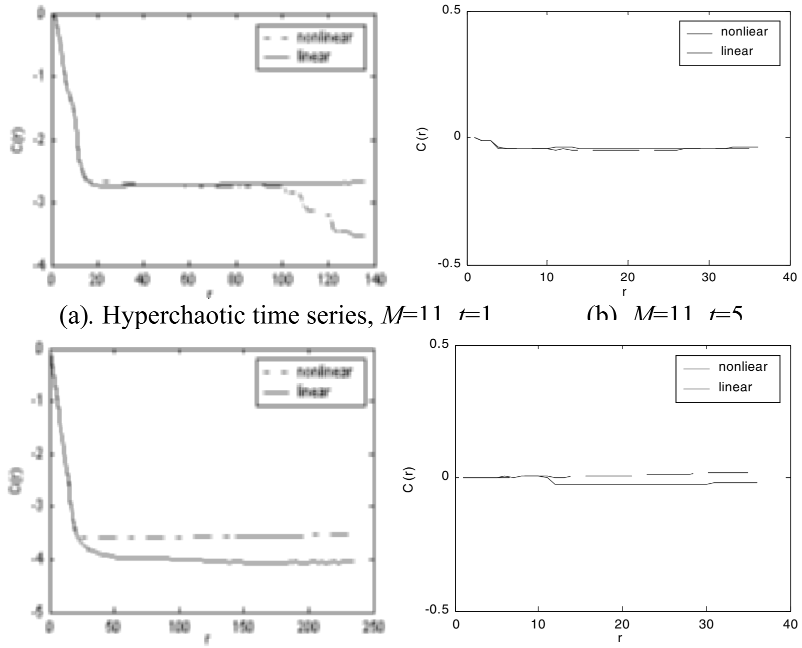
Sampling interval is as large as possible in the case of which can keep the synchronization of the crypto system. The effect of the sampling interval on the VWK test method is shown in Fig.1. When $\tau = 0.005$, most of the original data are linear, i.e. C^{nl} $(r) \approx C^{lin}(r)$. The initial time series has linear properties, which indicates that if the sampling interval is too small, it will be difficult to get the test result. However, the attackers can make use of this weakness to accomplish the linear reconstruction (linear modeling) (Hilborn and Ding 1996, Gibson et al 1992). If $\tau = 1$, the information of the linear model and nonlinear model are very similar, and they both are so small that we could assure that the original data have nonlinear properties. If we enlarge the sampling interval, the original data will be more like noise. In this case, according to the VWK test method, we can't figure out whether or not there are nonlinear elements in signals. Attackers can't identify if it is accurate signal or random noise either. If $\tau = 0.1$, it is obvious that C^{nl} (r) is less than $C^{lin}(r)$. In this situation, the initial series is nonlinear time series. However, this kind of the SI isn't good enough because it can be attacked by the nonlinear reconstruction method (attractor reconstruction) (Kennel et al. 1992, Palus and Dvorak 1992).

In the case of sub-Nyquist sampling interval, according to the VWK test method, we do not know that if there are accurate elements in the time series and the initial time series are much like random noises. From that point of view, we can use the VWK test method to analyze the time series of the chaotic system.



(a) $\tau = 0.005$ (b) $\tau = 1$ (c) $\tau = 0.1$

Fig. 1. The analysis effects of SI on VWK test



(a). Hyperchaotic time series, *M*=11 *t*=1 (b) *M*=11 *t*=5

(c). *M*=101, *t*=1 (d). *M*=101, *t*=5

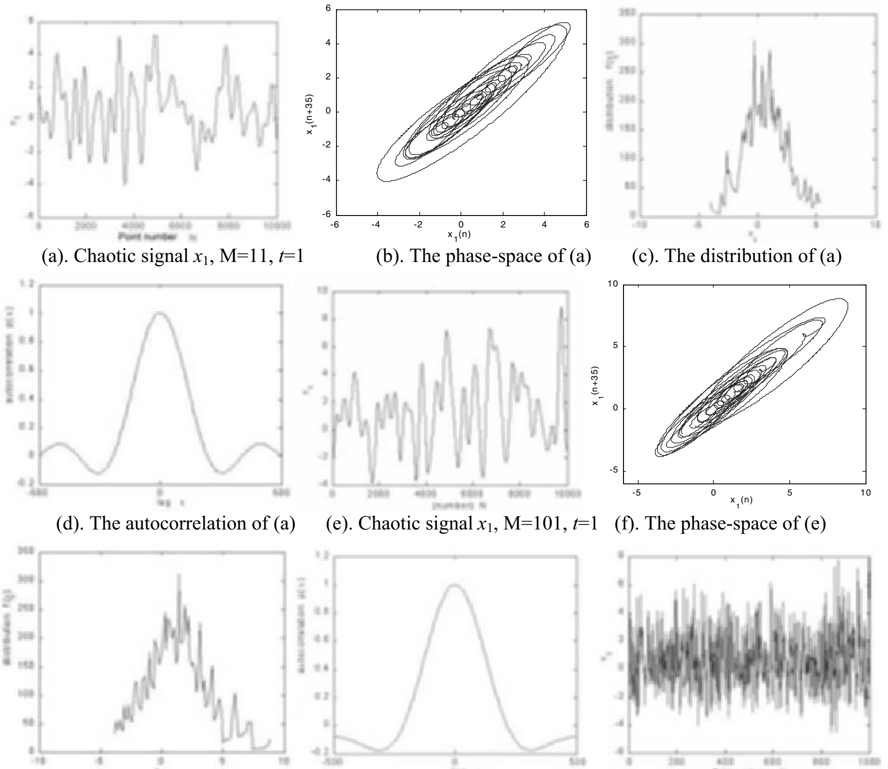
Fig.2. The VWK test analysis for the chaotic time series given by the chaotic encryption system

2.3 Application

Fig.2 (a) is the hyperchaotic system time series generated by formula (1) whose dimension is 11 and $t=1$ C^{nl} (r) is obviously less than $C^{lin}(r)$. Though it looks like random noise, the analysis result shows that it can be depicted using a nonlinear model and it has no security as we desire. Fig.2 (b) is the test result of the situation $t=5$. C^{nl} (r) and $C^{lin}(r)$ are both small. We also studied higher dimension (M=101) system. Fig.2 (c) shows the time series graph when $t=1$. Both C^{nl} (r) and $C^{lin}(r)$ are large, which means that the system is not secure. When $t=5$, C^{nl} (r) and $C^{lin}(r)$ become small, the pseudo-randomicity of the system is improved, as shown in Fig.2 (b) & (d). We can see that if the sampling interval is the same, the hyperchaotic system still put up the assured elements. Therefore, not only the dimension but also the sampling interval determines the security of the system. Because of the calculation ability, we only give out the greatest count results in this paper when $k \le 50$ and $d \le 3$.

3. The Cryptanalysis

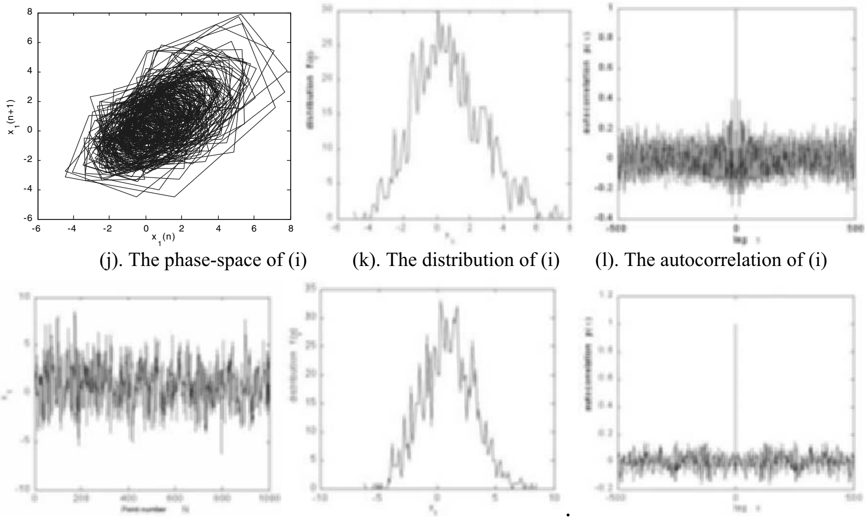
The time series analysis graphs for the hyperchaotic system (1) are shown in Fig. 3 (a)-(o). First, we study the situation with small a sampling interval. Let t=1, we can find from Fig.3 (a)-(h) that no matter how high is M, 11 or 101, there are no difference for the results. The chaotic curves have local linearzation charteristics. The phase space curves are very smooth and the corresponding δ specialties of the autocorrelation functions are bad. This indicates that if the SI is small, even if the hyperchaotic system is very complex, it is still menaced by the method of linear forecasting and the phase space reconstruction (Gibson et al 1992). In other words, this kind of chaotic system has no enough security. While, if we enlarge the SI, let t=5, the situation is totally different (see Fig.3 (i)-(p)). The autocorrelation property of the cryptosystem becomes better. In addition, with the same data length (N=1000), the higher the chaotic system dimension is, the better the autocorrelation property of the chaotic time series is, i.e., with the better encryption property. Fig.3 (c), (g), (k) and (n) show that the dimension M and SI do not have any effect on the distribution of the chaotic signal.



(a). Chaotic signal x_1, M=11, t=1 (b). The phase-space of (a) (c). The distribution of (a)

(d). The autocorrelation of (a) (e). Chaotic signal x_1, M=101, t=1 (f). The phase-space of (e)

(g). The distribution of (e) (h). The autocorrelation of (e) (i). Chaotic signal x_1, M=11, t=5



(j). The phase-space of (i) (k). The distribution of (i) (l). The autocorrelation of (i)

(m). Chaotic signal x_1, M=101,t=5 (n). The distribution of (m) (o). The autocorrelation of (m)

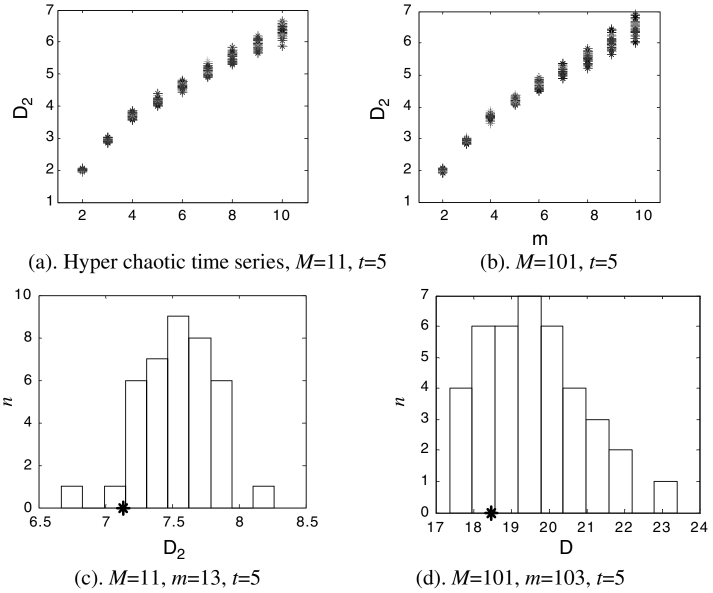
Fig. 3. The analysis of hyperchaotic system

Through the above analysis on the hyperchaotic system, we can conclude that the sub-Nyquist sampling interval is more suitable for chaotic systems to be encrypted as it has better encryption properties. We also use the surrogate data method to verify our conclusion. In certain condition, the pseudo- randomness of chaotic time series can be determined. Thus it can be used to analyze the pseudo-randomness of chaotic signal generated from the chaotic encryption system.

4. Nonlinear Test Study Based on the Surrogate Data

We choose 1000 points from the hyperchaotic system time series. If the sampling interval is $t = 5$ then according to the zero supposition in (Parlitz and Kocarev 1997) we can produce 39 sets of surrogate data by the algorithms in (Palus and Dvorak 1992). If the credibility is p then the surrogate data collection is $B_{\min} = 2/(1-p)-1$. Therefore, when $p = 95\%$ and $B_{\min} = 39$, the test statistics has the correlation dimension: $D = \lim\limits_{l \to 0} \frac{\ln C(l)}{\ln(l)}$.

For the hyperchaotic system with M=11dimensions (see Fig.4 (a)), if $m = 2 \sim 10$ then the surrogate data and the original data have no significant difference and the zero supposition is accepted. If m is bigger than 11, what will be the situation? Looking at Fig.4(c), m=13, we still can't find any difference between the original data and the surrogate data. This indicates that if the decrypter does not know the structure of the system it will be impossible to find useful information in the background noises. Even if the dimension is higher, M=101, the results are the same (see Fig.4 (b) and (d)). The vertical axes of the Fig.4(c) and Fig.4 (d) are the number of the surrogate data.



(a). Hyper chaotic time series, M=11, t=5

(b). M=101, t=5

(c). M=11, m=13, t=5

(d). M=101, m=103, t=5

Fig. 4. The results of chaotic encryption system by surrogate data test

5. Conclusion

In this paper, we first pointed out the disadvantages of the method discussed in (Parlitz and Kocarev 1997) and then proposed a new method to improve the security of the chaotic cryptosystems. Through studying the effect of the sampling interval on the VWK test, we found that when the SI is a sub-Nyquist sampling, the original data appear like noise apparently and they can express randomness in nature. It is therefore very difficult to depict the time series by a model method so that attackers can't decrypt information effectively. Based on the VWK test method, the analysis result for the hyperchaotic system referred in (Parlitz and Kocarev 1997) proved that if the sampling interval is too small the hyperchaotic system still has no enough security even though the dimension of the system is very large. The time series generated by the chaotic system has accurate elements. In addition, if the SI is too large the security of the hyperchaotic system is bad too. Only when the SI is the sub-Nyquist sampling, the chaotic time series is similar to be the random noise and the security will be high. We also used the surrogate data method to analyze the hyperchaotic system in (Parlitz and Kocarev 1997) in the sub-Nyquist sampling. In conclusion, when designing a chaotic cryptosystem with high security, the designer should consider not only adding the dimension of the system but also taking a suitable sampling interval.

Acknowledgement and Disclaimer

This work is supported by the National Natural Science Foundation of China (Grant No.50128504, 50575145). No approval or endorsement by the US National Institute of Standards and Technology is intended or implied.

References

Parlitz, U., Kocarev, L. (1997), Using surrogate data analysis for unmasking chaotic communication systems, International Journal of Bifurcation and Chaos, 7(2): 407-413

Parlitz, U.(1998) Nonlinear time series analysis, Proceedings of the 3rd International Specialist Workshop on Nonlinear Dynamics of Electronic Systems, p179-192

Abarbanel, H. D. I., Brown, R., Sidorowich, J. J. et al (1993), The analysis of observed chaotic data in physical systems, Review Modern Physics, 65(4): 1331-1392

Barahona, M., Poon, Shi-Sang (1996), Detection of nonlinear dynamics in short, noisy time series, Nature, 381: 215-217

Hilborn, R.C., Ding, M.Z. (1996), Optimal Reconstruction space for Estimating Correlation Dimension. International Journal of Bifurcation and Chaos, 6(2): 377~381

Gibson, J. F. Farmer, J. D., Casdagli, M. et al (1992), An analytic approach to practical state space reconstruction. Physica D, 57: 1-30.

Kennel, M., Brown R., and Abarbanel, H. D. I. (1992), Determining embedding dimension for phase-space reconstruction using a geometrical construction, Physical Review A, 45(5): 3403-3411

Palus, M. and Dvorak, I. (1992), Singular-value decomposition in attractor reconstruction: pitfalls and precautions, Physica D, 55: 221-234

Sakaguchi, Hidetsugu. (2002), Parameter evaluation from time sequences using chaos synchronization, Physical Review E, 65, p027201-1-4

Theiler, J., Prichard, D. (1996), Constrained-realization Monte- Carlo method for hypothesis testing, Physica D, 94: 221-235

Lei, M., Wang, Z. Z. and Feng, Z. J. (2001), Detecting nonlinearity action surface EMG signal, Physics Letters A, 290: 297-303.

Lei, M. (2002), Chaotic Time Series Analysis and Its Application Study on Chaotic Encryption System. PhD Thesis, Shanghai Jiao Tong University, China

Parlitz, U., Chua, L. O., Kocarev, Lj., Halle, K. S., and Shang, A. (1992), Transmission of digital signals by chaotic synchronization, International Journal of Bifurcation and Chaos, 2(4): 973-977

Parlitz, U., Kocarev, L., Stojanovski, T. & Preckel, H. (1996), Encoding messages using chaotic synchronization, Physical Review E53, 4351-4361

Pecora, L. M. & Carroll, T. L. (1990), Synchronization in chaotic systems, Physical Review Letter, 64, 821-824

Koc, Muammer and Lee, J. (2001), A system framework for next-generation e-maintenance systems, http://www.umw.edu/ceas/ims/

Subject Index

Index of Contributors