Leszek Rutkowski Rafał Scherer
Ryszard Tadeusiewicz Lotfi A. Zadeh
Jacek M. Zurada (Eds.)

LNAI 6114

Artificial Intelligence and Soft Computing

10th International Conference, ICAISC 2010
Zakopane, Poland, June 2010, Part II

2 Part II

Springer

Lecture Notes in Artificial Intelligence 6114

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

Leszek Rutkowski Rafał Scherer
Ryszard Tadeusiewicz Lotfi A. Zadeh
Jacek M. Zurada (Eds.)

Artifical Intelligence and Soft Computing

10th International Conference, ICAISC 2010
Zakopane, Poland, June 13-17, 2010, Part II

Springer

Volume Editors

Leszek Rutkowski
Częstochowa University of Technology, Poland
E-mail: lrutko@kik.pcz.czest.pl

Rafał Scherer
Częstochowa University of Technology, Poland
E-mail: rafal.scherer@kik.pcz.pl

Ryszard Tadeusiewicz
AGH University of Science and Technology, Kraków, Poland
E-mail: rtad@agh.edu.pl

Lotfi A. Zadeh
University of California, Berkeley, CA, USA
E-mail: zadeh@cs.berkeley.edu

Jacek M. Zurada
University of Louisville, KY, USA
E-mail: jacek.zurada@ louisville.edu

Library of Congress Control Number: 2010927691

CR Subject Classification (1998): I.2, H.3, F.1, I.4, H.4, H.5

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-642-13231-6 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-13231-5 Springer Berlin Heidelberg New York

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

springer.com

© Springer-Verlag Berlin Heidelberg 2010
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

This volume constitutes the proceedings of the 10th International Conference on Artificial Intelligence and Soft Computing, ICAISC 2010, held in Zakopane, Poland during June 13-17, 2010. The conference was organized by the Polish Neural Network Society in cooperation with the Academy of Management in Łódź (SWSPiZ), the Department of Computer Engineering at the Czestochowa University of Technology, and the IEEE Computational Intelligence Society, Poland Chapter. The previous conferences took place in Kule (1994), Szczyrk (1996), Kule (1997) and Zakopane (1999, 2000, 2002, 2004, 2006, 2008) and attracted a large number of papers and internationally recognized speakers: Lotfi A. Zadeh, Shun-ichi Amari, Daniel Amit, Piero P. Bonissone, Zdzislaw Bubnicki, Andrzej Cichocki, Wlodzislaw Duch, Pablo A. Estévez, Jerzy Grzymala-Busse, Kaoru Hirota, Janusz Kacprzyk, Laszlo T. Koczy, Soo-Young Lee, Robert Marks, Evangelia Micheli-Tzanakou, Erkki Oja, Witold Pedrycz, Sarunas Raudys, Enrique Ruspini, Jorg Siekman, Roman Slowinski, Ryszard Tadeusiewicz, Shiro Usui, Ronald Y. Yager, Syozo Yasui and Jacek Zurada. The aim of this conference is to build a bridge between traditional artificial intelligence techniques and recently developed soft computing techniques. It was pointed out by Lotfi A. Zadeh that "Soft Computing (SC) is a coalition of methodologies which are oriented toward the conception and design of information/intelligent systems. The principal members of the coalition are: fuzzy logic (FL), neurocomputing (NC), evolutionary computing (EC), probabilistic computing (PC), chaotic computing (CC), and machine learning (ML). The constituent methodologies of SC are, for the most part, complementary and synergistic rather than competitive." This volume presents both traditional artificial intelligence methods and soft computing techniques. Our goal is to bring together scientists representing both traditional artificial intelligence approaches and soft computing techniques. This volume is divided into four parts:

- Fuzzy Systems and Their Applications
- Data Mining, Classification and Forecasting
- Image and Speech Analysis
- Bioinformatics and Medical Applications

The conference attracted a total of 385 submissions from 44 countries and after the review process, 169 papers were accepted for publication. I would like to thank our participants, invited speakers and reviewers of the papers for their scientific and personal contribution to the conference. several reviewers were very helpful in reviewing the papers and are listed later.

Finally, I thank my co-workers Łukasz Bartczuk, Agnieszka Cpałka, Piotr Dziwiński, Marcin Gabryel, Marcin Korytkowski and the conference secretary Rafał Scherer for their enormous efforts to make the conference a very successful event. Moreover, I would appreciate the work of Marcin Korytkowski, who designed the Internet submission system.

June 2010 Leszek Rutkowski

Organization

ICAISC 2010 was organized by the Polish Neural Network Society in cooperation with the Academy of Management in Łódź (SWSPiZ), the Department of Computer Engineering at Częstochowa University of Technology, and the IEEE Computational Intelligence Society, Poland Chapter.

Program Chairs

Honorary Chair	Lotfi Zadeh (USA)
	Jacek Żurada (USA)
General Chair	Leszek Rutkowski (Poland)
Co-Chairs	Włodzisław Duch (Poland)
	Janusz Kacprzyk (Poland)
	Józef Korbicz (Poland)
	Ryszard Tadeusiewicz (Poland)

Program Committee

Rafał Adamczak, Poland
Cesare Alippi, Italy
Shun-ichi Amari, Japan
Rafal A. Angryk, USA
Jarosław Arabas, Poland
Robert Babuska, The Netherlands
Ildar Z. Batyrshin, Russia
James C. Bezdek, USA
Leon Bobrowski, Poland
Leonard Bolc, Poland
Piero P. Bonissone, USA
Bernadette Bouchon-Meunier, France
James Buckley, Poland
Tadeusz Burczynski, Poland
Andrzej Cader, Poland
Juan Luis Castro, Spain
Yen-Wei CHEN, Japan
Wojciech Cholewa, Poland
Fahmida N. Chowdhury, USA
Andrzej Cichocki, Japan
Paweł Cichosz, Poland
Krzysztof Cios, USA
Ian Cloete, Germany
Oscar Cordón, Spain
Bernard De Baets, Belgium
Nabil Derbel, Tunisia
Ewa Dudek-Dyduch, Poland
Ludmiła Dymowa, Poland
Andrzej Dzieliński, Poland
David Elizondo, UK
Meng Joo Er, Singapore
Pablo Estevez, Chile
János Fodor, Hungary
David B. Fogel, USA
Roman Galar, Poland
Alexander I. Galushkin, Russia
Adam Gaweda, USA
Joydeep Ghosh, USA
Juan Jose Gonzalez de la Rosa, Spain
Marian Bolesław Gorzałczany, Poland
Krzysztof Grąbczewski , Poland
Garrison Greenwood, USA
Jerzy W. Grzymala-Busse, USA
Hani Hagras, UK
Saman Halgamuge, Australia
Rainer Hampel, Germany

Zygmunt Hasiewicz, Poland
Yoichi Hayashi, Japan
Tim Hendtlass, Australia
Francisco Herrera, Spain
Kaoru Hirota, Japan
Adrian Horzyk, Poland
Tingwen Huang, USA
Hisao Ishibuchi, Japan
Mo Jamshidi, USA
Andrzej Janczak, Poland
Norbert Jankowski, Poland
Robert John, UK
Jerzy Józefczyk, Poland
Tadeusz Kaczorek, Poland
Władysław Kamiński, Poland
Nikola Kasabov, New Zealand
Okyay Kaynak, Turkey
Vojislav Kecman, New Zealand
James M. Keller, USA
Etienne Kerre, Belgium
Frank Klawonn, Germany
Jacek Kluska, Poland
Leonid Kompanets, Poland
Przemysław Korohoda, Poland
Jacek Koronacki, Poland
Witold Kosiński, Poland
Jan M. Kościelny, Poland
Zdzisław Kowalczuk, Poland
Robert Kozma, USA
László Kóczy, Hungary
Rudolf Kruse, Germany
Boris V. Kryzhanovsky, Russia
Adam Krzyzak, Canada
Juliusz Kulikowski, Poland
Roman Kulikowski, Poland
Věra Kůrková, Czech Republic
Marek Kurzyński, Poland
Halina Kwaśnicka, Poland
Soo-Young Lee, Korea
George Lendaris, USA
Antoni Ligęza, Poland
Zhi-Qiang LIU, Hong Kong
Simon M. Lucas, UK
Jacek Łęski, Poland
Bohdan Macukow, Poland
Kurosh Madani, France
Luis Magdalena, Spain
Witold Malina, Poland
Krzysztof Malinowski, Poland
Jacek Mańdziuk, Poland
Antonino Marvuglia, Ireland
Andrzej Materka, Poland
Jaroslaw Meller, Poland
Jerry M. Mendel, USA
Radko Mesiar, Slovakia
Zbigniew Michalewicz , Australia
Zbigniew Mikrut, Poland
Sudip Misra, USA
Wojciech Moczulski, Poland
Javier Montero, Spain
Eduard Montseny, Spain
Detlef D. Nauck, Germany
Antoine Naud, Poland
Edward Nawarecki, Poland
Antoni Niederliński, Poland
Robert Nowicki, Poland
Andrzej Obuchowicz, Poland
Marek Ogiela, Poland
Erkki Oja, Finland
Stanisław Osowski, Poland
Nikhil R. Pal, India
Maciej Patan, Poland
Witold Pedrycz, Canada
Leonid Perlovsky, USA
Andrzej Pieczyński, Poland
Andrzej Piegat, Poland
Vincenzo Piuri, Italy
Lech Polkowski, Poland
Marios M. Polycarpou, Cyprus
Danil Prokhorov, USA
Anna Radzikowska, Poland
Ewaryst Rafajłowicz, Poland
Sarunas Raudys, Lithuania
Olga Rebrova, Russia
Vladimir Red'ko, Russia
Raúl Rojas, Germany
Imre J. Rudas, Hungary
Enrique H. Ruspini, USA
Khalid Saeed, Poland
Dominik Sankowski, Poland

Norihide Sano, Japan
Robert Schaefer, Poland
Rudy Setiono, Singapore
Paweł Sewastianow, Poland
Jennie Si, USA
Peter Sincak, Slovakia
Andrzej Skowron, Poland
Ewa Skubalska-Rafajłowicz, Poland
Roman Słowiński, Poland
Tomasz G. Smolinski, USA
Czesław Smutnicki, Poland
Pilar Sobrevilla, Spain
Jerzy Stefanowski, Poland
Pawel Strumillo, Poland
Ron Sun, USA
Johan Suykens Suykens, Belgium
Piotr Szczepaniak, Poland
Eulalia J. Szmidt, Poland
Przemysław Śliwiński, Poland
Adam Słowik, Poland
Jerzy Świątek , Poland
Hideyuki Takagi, Japan
Yury Tiumentsev, Russia
Vicenç Torra, Spain
Burhan Turksen, Canada
Shiro Usui, Japan
Michael Wagenknecht, Germany
Tomasz Walkowiak, Poland
Deliang Wang, USA
Jun Wang, Hong Kong
Lipo Wang, Singapore
Zenon Waszczyszyn, Poland
Paul Werbos, USA
Slawo Wesolkowski, Canada
Sławomir Wiak, Poland
Bernard Widrow, USA
Kay C. Wiese, Canada
Bogdan M. Wilamowski, USA
Donald C. Wunsch, USA
Maciej Wygralak, Poland
Roman Wyrzykowski, Poland
Ronald R. Yager, USA
Gary Yen, USA
John Yen, USA
Sławomir Zadrożny, Poland
Ali M. S. Zalzala, United Arab Emirates

Organizing Committee

Rafał Scherer, Secretary
Łukasz Bartczuk, Organizing Committee Member
Piotr Dziwiński, Organizing Committee Member
Marcin Gabryel, Organizing Committee Member
Marcin Korytkowski, Databases and Internet Submissions

External Reviewers

R. Adamczak
J. Arabas
T. Babczyński
Ł. Bartczuk
A. Bielecki
A. Bielskis
J. Biesiada
M. Blachnik
L. Bobrowski
P. Boguś
M. Borawski
A. Borkowski
W. Bozejko
T. Burczyński
R. Burduk
B. Butkiewicz
C. Castro
K. Cetnarowicz
M. Chang
M. Chis
W. Cholewa
R. Choraś
A. Cichocki
P. Cichosz
R. Cierniak
S. Concetto
B. Cyganek
R. Czabański
I. Czarnowski
B. De Baets

K. Delac
V. Denisov
G. Dobrowolski
A. Dzieliński
P. Dziwiński
S. Ehteram
D. Elizondo
M. Flasiński
C. Frowd
M. Gabryel
A. Gawęda
M. Giergiel
F. Gomide
M. Gorzałczany
K. Grąbczewski
K. Grudziński
J. Grzymala-Busse
P. Hajek
Z. Hasiewicz
Y. Hayashi
O. Henniger
F. Herrera
Z. Hippe
A. Horzyk
M. Hrebień
A. Janczak
N. Jankowski
J. Jelonkiewicz
J. Kacprzyk
W. Kamiński
A. Kasperski
V. Kecman
E. Kerre
F. Klawonn
L. Koczy
J. Konopacki
J. Korbicz
P. Korohoda
J. Koronacki
M. Korytkowski
M. Korzeń
J. Kościelny
L. Kotulski
Z. Kowalczuk
J. Kozlak
M. Kretowski
R. Kruse
B. Kryzhanovsky
A. Krzyzak
J. Kulikowski
V. Kurkova
M. Kurzyński
H. Kwaśnicka
A. Ligęza
J. Łęski
K. Madani
W. Malina
J. Mańdziuk
U. Markowska-Kaczmar
A. Marvuglia
A. Materka
J. Mendel
R. Mesiar
Z. Michalewicz
J. Michalkiewicz
Z. Mikrut
W. Mokrzycki
E. Nawarecki
M. Nieniewski
A. Niewiadomski
R. Nowicki
A. Obuchowicz
S. Osowski
A. Owczarek
F. Pappalardo
K. Patan
W. Pedrycz
A. Pieczyński
Z. Pietrzykowski
V. Piuri
T. Przybyła
E. Rafajłowicz
R. Rojas
L. Rolka
I. Rudas
M. Rudnicki
L. Rutkowski
R. Schaefer
R. Scherer
R. Setiono
A. Sędziwy
W. Skarbek
A. Skowron
E. Skubalska-Rafajłowicz
K. Slot
A. Słowik
R. Słowiński
T. Smolinski
C. Smutnicki
J. Starczewski
P. Strumiłło
J. Swacha
E. Szmidt
P. Śliwiński
J. Świątek
R. Tadeusiewicz
H. Takagi
Y. Tiumentsev
V. Torra
J. Verstraete
M. Wagenknecht
T. Walkowiak
J. Wang
L. Wang
S. Wiak
B. Wilamowski
P. Wojewnik
M. Wygralak
W. Xu
F. Zacarias
S. Zadrożny
J. Zieliński

Table of Contents – Part II

Part I: Neural Networks and Their Applications

Part II: Evolutionary Algorithms and Their Applications

Part III: Agent Systems, Robotics and Control

Part IV: Various Problems of Artificial Intelligence

Table of Contents – Part I

Part I: Fuzzy Systems and Their Applications

Part II: Data Mining, Classification and Forecasting

Part III: Image and Speech Analysis

Part IV: Bioinformatics and Medical Applications

Part I

Neural Networks and Their Applications

Complex-Valued Neurons with Phase-Dependent Activation Functions

Igor Aizenberg

Department of Computer Science,
Texas A&M University-Texarkana,
P.O. Box 5518, Texarkana, TX 75505-5518, USA
igor.aizenberg@tamut.edu

Abstract. In this paper, we observe two artificial neurons with complex-valued weights. There are a multi-valued neuron and a universal binary neuron. Both neurons have activation functions depending on the argument (phase) of the weighted sum. A multi-valued neuron may learn multiple-valued threshold functions. A universal binary neuron may learn arbitrary (not only linearly-separable) Boolean functions. It is shown that a multi-valued neuron with a periodic activation function may learn non-threshold functions by their projection to the space corresponding to the larger valued logic. A feedforward neural network with multi-valued neurons and its learning are also considered.

Keywords: complex-valued neural networks, derivative-free learning, multi-valued neuron.

1 Introduction

Complex-valued neural networks become increasingly popular. The use of complex-valued inputs/outputs, weights and activation functions make it possible to increase the functionality of a single neuron and of a neural network, to improve their performance and to reduce the training time [1]. There are different specific types of complex-valued neurons and complex-valued activation functions [1]. We will consider here two neurons that have been proposed first among complex-valued neurons, a multi-valued neuron (MVN) and a Universal Binary Neuron (UBN). This is an overview paper based on work published recently by the author (sole and in collaboration with other colleagues), and on works published by other researchers. Specifically we focus on [2-5]. The main concepts behind both mentioned neurons are use of complex-valued weights, location of neurons inputs/outputs on the unit circle, and activation functions, which depend on the argument (phase) of the weighted sum. These ideas were introduced in by N. Aizenberg et. al. in their seminal paper [6], where a concept of multiple-valued logic over the field of complex numbers and the first complex-valued activation function were proposed. The discrete MVN was introduced in [7]. This neuron is based on the concept of multiple-valued logic over the field of complex numbers, which was introduced in [6], then presented in detail in 8], and further developed in [9]. The new approach

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 3–10, 2010.
© Springer-Verlag Berlin Heidelberg 2010

significantly increased the functionality of a single neuron (compared to the traditional sigmoidal neuron) and simplified its learning. The continuous MVN was introduced in [10] and then developed in [2]. Recently, a periodic activation function was suggested for the MVN in [5]. This function increases the MVN's functionality even more, making it possible to learn *k*-valued functions, which are not threshold, using a single neuron.

The UBN was introduced in [11], then its theory was presented in detail in [9], and its modified learning algorithm is considered in [4]. The UBN is a Boolean neuron, its inputs and output are binary, thus it implements those input/output mappings, which are Boolean functions. The most important advantage of the UBN over traditional neurons is its ability to learn nonlinearly separable Boolean functions.

2 Multi-Valued Neuron (MVN)

2.1 Discrete and Continuous MVN

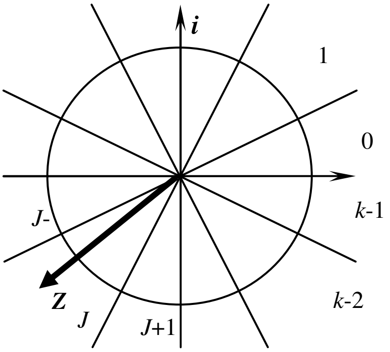
The discrete MVN was proposed in [7]. It is based on the principles of multiple-valued threshold logic over the field of complex numbers. The discrete MVN performs a mapping between *n* inputs and a single output. This mapping is described by a multiple-valued (*k*-valued) threshold function of *n* variables $f(x_1, \ ..., \ x_n)$. It is important to specify that we consider here multiple-valued logic over the field of complex numbers [8, 9]. While in traditional multiple-valued logic its values are encoded by the integers from the set $K = \{0, 1, ..., k-1\}$, in the one over the field of complex numbers they are encoded by the *k*th roots of unity $E_k = \{\varepsilon^0, \varepsilon, \varepsilon^2, ..., \varepsilon^{k-1}\}$, where $\varepsilon^j = e^{i2\pi j/k}$, $j = 0, ..., k - 1$, (*i* is an imaginary unity). A *k*-valued threshold function describing the MVN input/output mapping, can be represented using $n+$ 1 complex-valued weight as follows

$$f(x_1, ..., x_n) = P(w_0 + w_1 x_1 + ... + w_n x_n), \tag{1}$$

where $x_1, ..., x_n$ are the variables (neuron inputs) and $w_0 ,w_1 , ...,w_n$ are the weights. The values of the function and of the neuron inputs are the k^{th} roots of unity: $\varepsilon^j = e^{i2\pi j/k}$, $j \in \{0,1,...,k - 1\}$, *i* is an imaginary unity. *P* is the activation function

$$P(z) = e^{i2\pi j/k}, \text{ if } 2\pi j / k \le \arg z < 2\pi (j+1) / k, \tag{2}$$

where *j*=0, 1, ..., *k*-1 are values of *k*-valued logic, $z = w_0 + w_1 x_1 + ... + w_n x_n$ is the weighted sum, arg *z* is the argument of the complex number *z*. It is important to mention that function (2), which was introduced in [6], is historically the first known complex-valued activation function. Function (2) divides a complex plane into *k* equal sectors and maps the whole complex plane into a set of k^{th} roots of unity (see Fig. 1).



$$P(z)=\exp\left(j\cdot i2\pi/k\right)$$

Fig. 1. Geometrical interpretation of the discrete MVN activation function

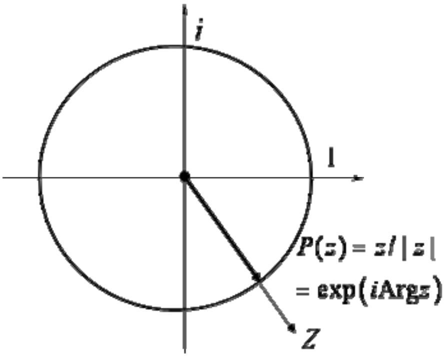


Fig. 2. Geometrical interpretation of the continuous MVN activation function

The continuous MVN has been proposed in [10] and then developed in [2]. The continuous case corresponds to $k \to \infty$ in (2). If the number of sectors $k \to \infty$ (see Fig. 1), then the angular size of a sector tends to 0. Hence, an activation function in this case becomes simply a projection of the weighted sum $z = w_0 + w_1x_1 + ... + w_nx_n$ onto the unit circle:

$$P(z) = \exp(i\ (\arg z)) = e^{iArg\ z} = z/\mid z\mid, \tag{3}$$

where z is the weighted sum, Arg z is a main value of its argument (phase) and $|z|$ is the absolute value of the complex number z. Activation function (3) is illustrated in Fig. 2. It maps the whole complex plane into the unit circle. Evidently, a hybrid MVN (with continuous inputs/discrete output, discrete inputs/continuous output, hybrid inputs and discrete or continuous output can also be easily considered.

2.2 MVN Learning

MVN learning algorithm is identical for both discrete and continuous neurons. The most important property of MVN learning is that it is derivative-free. There are two MVN learning algorithms. One of them, which was proposed in [9], is computationally more efficient. It is based on the error-correction learning rule. If T is the desired neuron's output and Y is the actual one, then $\delta = T - Y$ is the error (see Fig. 3) that determines that adjustment of the weights, which is performed as follows:

$$W_{r+1} = W_r + \frac{C_r}{(n+1)}\delta\overline{X}\ , \tag{4}$$

or, as it was suggested in [2], as follows

$$W_{r+1} = W_r + \frac{C_r}{(n+1)|z_r|}\delta \overline{X}\ , \tag{5}$$

where $\overline{X}$ is the vector of inputs with the components complex-conjugated, n is the number of neuron inputs, δ is the neuron's error, r is the number of the learning iteration, W_r is the current weighting vector (to be corrected), W_{r+1} is the following weighting vector (after correction), C_r is the constant part of the learning rate (it may always be equal to 1), and $|z_r|$ is the absolute value of the weighted sum obtained on the r^{th} iteration. A factor $1/|z_r|$ in (5) is a variable part of the learning rate. The use of it can be important for learning highly nonlinear input/output mappings.

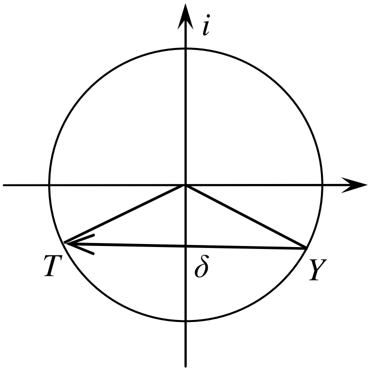


Fig. 3. Geometrical interpretation of the MVN learning rule

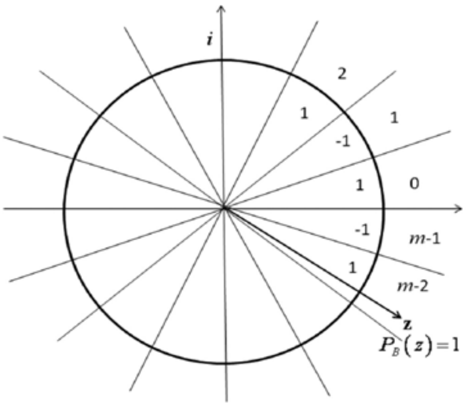


Fig. 4. Geometrical interpretation of the UBN activation function

2.3 Multilayer Neural Network with Multi-Valued Neurons (MLMVN)

Multiple applications of MVN have been proposed including cellular neural networks [9], associative memories [7], [9], [12-17], classifiers [15], [17], [18], and multi-valued nonlinear filter (MVF) [9] One of the most impressive applications of MVN is a multilayer neural network based on multi-valued neurons (MLMVN).

It was suggested in [10] to use the MVN as a basic neuron in a feedforward neural network. This network (MLMVN), which can consist of both continuous and discrete MVNs, and its derivative-free backpropagation learning algorithm were explicitly presented in [2]. This learning algorithm was generalized in [3] for a network with an arbitrary amount of output neurons. The MLMVN learns faster and generalizes better than a traditional feedforward network and different kernel-based networks when solving both benchmark and real world problems [2], [3].

The MLMVN's topology is identical to the one of classical feedforward neural networks. The network consists of *m* layers: one input layer, *m*-1 hidden layers and one output layer. The outputs of each neuron from a given layer are connected to the corresponding inputs of each neuron from the following layer. A backpropagation learning algorithm for the MLMVN is derivative-free. It is shown in [2], [3] that the MLMVN learning (as well as the MVN learning) depends only on the global error of the network and the adjustment of the weights is also determined only by this error. Given that T_{km} is a desired output of the *k*th neuron from the output layer and Y_{km} is an actual output of the *k*th neuron from the output layer, the global error of the network taken from the *k*th neuron of the *m*th (output) layer is calculated as follows:

$$\delta^*_{km} = T_{km} - Y_{km} . \tag{6}$$

Based on such formulation of error, backpropagation learning algorithm for the MLMVN has been proposed in [2]. Let w_i^{kj} be the weight corresponding to the *i*th input of the *kj*th neuron (*k*th neuron of the *j*th layer), Y_{ij} be the actual output of the *i*th neuron from the *j*th layer (*j*=1,…,*m*), and N_j be the number of the neurons in the *j*th layer. Let $x_1,...,x_n$ be the inputs of the network. The errors of the neurons from layers 1 to *m*-1 are obtained according to (7) and the errors of the neurons from the output (*m*-th) layer are obtained according to (8):

$$\delta_{kj} = \frac{1}{s_j}\sum_{i=1}^{N_{j+1}} \delta_{ij+1}(w_k^{ij+1})^{-1} , \tag{7}$$

$$\delta_{km} = \frac{1}{s_m}\delta^*_{km} \tag{8}$$

In (7) and (8), *kj* indicates the *k*th neuron in the *j*th layer and $s_j = N_{j-1} + 1$ is the number of all neurons in layer *j*-1 incremented by 1 ($s_1 = 1$). Given errors assigned to each neuron, the weights of neurons are adjusted using the learning rule (4) for the hidden neurons and (5) for the output neurons. The MLMVN training can stop when the mean square error reaches a certain low level.

3 Universal Binary Neuron (UBN) and Multi-Valued Neuron with a Periodic Activation Function (MVN-P)

The UBN was introduced in [11]. It is presented in detail in [9], and its modified learning algorithm was recently considered in [4]. A key idea behind the UBN is the use of complex-valued weights and an original activation function for learning nonlinearly separable Boolean functions.

If k=2 in (2) then the complex domain is divided into two parts as well (the top semiplane (“1”) and the bottom semiplane (“-1”):

$$P(z)=\begin{cases} 1, & 0\le \arg(z)<\pi \\ -1, & \pi\le \arg(z)<2\pi. \end{cases}$$

However, this activation function does not increase the neuron’s functionality: although the weights are complex, the neuron still can only learn linearly separable Boolean functions. It was suggested in [11] to use an l-repetitive activation function

$$P_B(z)=(-1)^j,\text{if } 2\pi j/m\le \arg(z)<2\pi(j+1)/m;\ m=2l, l\in \mathbf{N}, \tag{9}$$

where l is some positive integer, j is a non-negative integer $0\le j<m$. Activation function (9) is illustrated in Fig. 4. It separates the complex plane into m =2l equal sectors and determines the neuron’s output by the alternating sequence of 1, -1, 1, -1,…1, -1 depending on the parity of the sector’s number.

The most important advantage of the UBN is its ability to learn nonlinearly separable Boolean functions. XOR and Parity n become nearly the simplest problems, which can be learned by a single UBN. For example, Table 1 shows how the XOR problem can be solved using a single UBN

Table 1. Solving the **XOR** problem using a single UBN with the weighting vector (0, i, 1)

#	x_1	x_2	$z=w_0+w_1x_1+w_2x_2$	$P_B(z)$	XOR== $x_1 \underset{\text{mod}\,2}{\oplus} x_2$
1)	**1**	**1**	$1+i$	**1**	**1**
2)	**1**	**-1**	$-1+i$	**-1**	**-1**
3)	**-1**	**1**	$1-i$	**-1**	**-1**
4)	**-1**	**-1**	$-1-i$	**1**	**1**

with the activation function (9) ($l=2, m=4$).The UBN learning algorithm is presented in detail in [4]. It is based on the same learning rules (4) and (5) as the MVN learning algorithm. The choice of the desired output is based on the closeness of the current weighted sum to the right or left adjacent sector.

The multi-valued neuron with a periodic activation function (MVN-P) was recently introduced in [5]. This activation function is defined as follows:

$$\begin{gathered} P_l(z)=j \bmod k,\ \text{if } 2\pi j/m\le \arg\ z<2\pi\,(j+1)/m, \\ j=0,1,\ldots,m-1;\ m=kl, l\ge 2. \end{gathered} \tag{10}$$

It is illustrated in Fig. 5. Evidently, for k=2 function (10) coincides with function (9) and in this case the MVN-P coincides with the UBN.

The most important property of the MVN-P is its ability to learn those k-valued input/output mappings, which are not threshold in k-valued logic and cannot be learned using a single MVN with activation function (2), but which are partially defined threshold functions in m-valued logic.

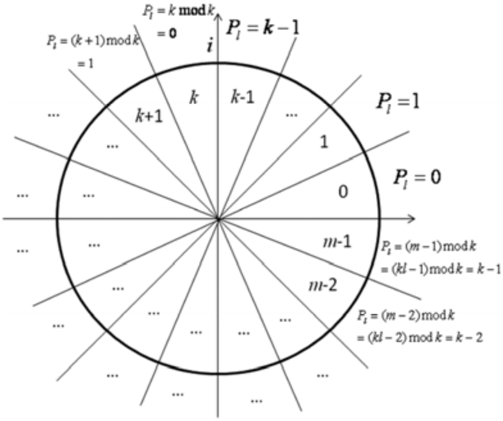


Fig. 5. Geometrical interpretation of the periodic activation function (10)

For example, such a multiple-valued function, as a mod k addition of n k-valued variables (a multiple-valued analog of the Parity n function) cannot be learned by a single MVN with activation function (2), but can be learned by a single MVN-P with activation function (10) [5]. The MVN-P learning algorithm is similar to the UBN learning algorithm [5]. Suppose that the current MVN-P's output is not correct and the current weighted sum is located in the sector $s \in M = \{0,1,...,m-1\}$. There are l sectors on the complex plane, where function (10) takes the correct value. Two of them are the closest ones to sector s (from right and left sides). A sector whose border is closest to the current weighted sum z is chosen as a desired sector q. Then either of learning rules (4) or (5) can be applied.

4 Conclusions

We observed in this paper in detail complex-valued neurons with the phase-dependent activation functions – the multi-valued neuron and the universal binary neuron.

Acknowledgement

This work is supported by the National Science Foundation under Grant 0925080.

References

1. Hirose, A.: Complex-Valued Neural Networks. Springer, Heidelberg (2006)
2. Aizenberg, I., Moraga, C.: Multilayer Feedforward Neural Network Based on Multi-Valued Neurons (MLMVN) and a Backpropagation Learning Algorithm. Soft Computing 11(2), 169–183 (2007)
3. Aizenberg, I., Paliy, D., Zurada, J.M., Astola, J.: Blur Identification by Multilayer Neural Network based on Multi-Valued Neurons. IEEE Transactions on Neural Networks 19(5), 883–898 (2008)
4. Aizenberg, I.: Solving the XOR and Parity n Problems Using a Single Universal Binary Neuron. Soft Computing 12(3), 215–222 (2008)

5. Aizenberg, I.: A Multi-Valued Neuron with a Periodic Activation Function. In: International Joint Conference on Computational Intelligence, Funchal-Madeira, Portugal, October 5-7, pp. 347–354 (2009)
6. Aizenberg, N.N., Ivaskiv, Y.L., Pospelov, D.A.: About one generalization of the threshold function Doklady Akademii Nauk SSSR (The Reports of the Academy of Sciences of the USSR), vol. 196(6), pp. 1287–1290 (1971) (in Russian)
7. Aizenberg, N.N., Aizenberg, I.N.: CNN Based on Multi-Valued Neuron as a Model of Associative Memory for Gray-Scale Images. In: The Second IEEE Int. Workshop on Cellular Neural Networks and their Applications, October 1992, pp. 36–41. Technical University Munich, Germany (1992)
8. Aizenberg, N.N., Ivaskiv, Y.L.: Multiple-Valued Threshold Logic. Naukova Dumka Publisher House, Kiev (1977) (in Russian)
9. Aizenberg, I., Aizenberg, N., Vandewalle, J.: Multi-valued and universal binary neurons: theory, learning, applications. Kluwer Academic Publishers, Dordrecht (2000)
10. Aizenberg, I., Moraga, C., Paliy, D.: A Feedforward Neural Network based on Multi-Valued Neurons. In: Reusch, B. (ed.) Computational Intelligence, Theory and Applications. Advances in Soft Computing, vol. XIV, pp. 599–612. Springer, Heidelberg (2005)
11. Aizenberg, I.N.: A Universal Logic Element over the Complex Field. Kibernetika Cybernetics and Systems Analysis 27(3), 116–121; English version is available from Springer 27(3), 467-473 (1991) (in Russian)
12. Jankowski, S., Lozowski, A., Zurada, J.M.: Complex-Valued Multistate Neural Associative Memory. IEEE Trans. Neural Networks 7(6), 1491–1496 (1996)
13. Aoki, H., Kosugi, Y.: An Image Storage System Using Complex-Valued Associative Memory. In: 15th International Conference on Pattern Recognition, Barcelona, vol. 2, pp. 626–629. IEEE Computer Society Press, Los Alamitos (2000)
14. Muezzinoglu, M.K., Guzelis, C., Zurada, J.M.: A New Design Method for the Complex-Valued Multistate Hopfield Associative Memory. IEEE Trans. Neural Networks 14(4), 891–899 (2003)
15. Aoki, H., Watanabe, E., Nagata, A., Kosugi, Y.: Rotation-Invariant Image Association for Endoscopic Positional Identification Using Complex-Valued Associative Memories. In: Mira, J., Prieto, A.G. (eds.) IWANN 2001. LNCS, vol. 2085, pp. 369–374. Springer, Heidelberg (2001)
16. Lee, D.L.: Improving the capacity of complex-valued neural networks with a modified gradient descent learning rule. IEEE Transactions on Neural Networks 12(2), 439–443 (2001)
17. Aoki, H.: A complex-valued neuron to transform gray level images to phase information. In: Wang, L., Rajapakse, J.C., Fukushima, K., Lee, S.-Y., Yao, X. (eds.) 9th International Conference on Neural Information Processing (ICONIP 2002), vol. 3, pp. 1084–1088 (2002)
18. Aizenberg, I., Myasnikova, E., Samsonova, M., Reinitz, J.: Temporal Classification of Drosophila Segmentation Gene Expression Patterns by the Multi-Valued Neural Recognition Method. Mathematical Biosciences 176(1), 145–159 (2002)

ART-Type Artificial Neural Networks Applications for Classification of Operational States in Wind Turbines*

Tomasz Barszcz[1], Andrzej Bielecki[2], and Mateusz Wójcik[3]

[1] Chair of Robotics and Mechatronics
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
[2] Institute of Computer Science,
Jagiellonian University,
Łojasiewicza 6, 30-348 Kraków, Poland
[3] Department of Computer Design and Graphics,
Jagiellonian University,
Reymonta 4, 30-059 Kraków, Poland
tbarszcz@agh.edu.pl, bielecki@ii.uj.edu.pl, mateusz.wojcik@uj.edu.pl

Abstract. In recent years wind energy is the fastest growing branch of the power generation industry. The largest cost for the wind turbine is its maintenance. A common technique to decrease this cost is a remote monitoring based on vibration analysis. Growing number of monitored turbines requires an automated way of support for diagnostic experts. As full fault detection and identification is still a very challenging task, it is necessary to prepare an "early warning" tool, which would focus the attention on cases which are potentially dangerous. There were several attempts to develop such tools, in most cases based on various classification methods (predominantly neural networks). Due to very common lack of sufficient data to perform training of a method, the important problem is the need for creation of new states when there are data different from all known states.

As the ART neural networks are capable to perform efficient classification and to recognize new states when necessary, they seems to be a proper tool for classification of operational states in wind turbines. The verification of ART and fuzzy-ART networks efficiency in this task is the topic of this paper.

1 Introduction

In recent years wind energy is the fastest growing branch of the power generation industry. The average yearly growth in the years 1997-2003 achieved 32% in the United States and 22% in the European Union [2] and these figures will hold for at least the next decade. The distribution of costs during the life cycle of

* The paper was supported by the Polish Ministry of Science and Higher Education under Grant No. N504 147838.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 11–18, 2010.
© Springer-Verlag Berlin Heidelberg 2010

the unit for wind energy is significantly different from that of traditional, fossil fired units [2]. First of all, initial investment costs are relatively higher, whereas in traditional units cost of fuel plays important role (usually it is the second largest cost). After commissioning, the largest cost for the wind turbine (WT for abbreviation) is maintenance. With proper maintenance policies, wind turbines can achieve the highest level of availability in the power generation sector - even up to 98%. The basis of proper maintenance is continuous monitoring of the transmission of the wind turbine. There were several attempts to develop various classification tools, in most cases based on various classification methods (predominantly neural networks). Due to very common lack of sufficient data to perform training of a method, the important problem is the need for creation of new states when there are data different from all known states.

In most types of artificial neural networks (ANNs for abbreviation) the learning process is unsuitable for cases of continuous machinery intelligence monitoring. This means, among others, that adding a new patterns as inputs requires repetition of the learning process. In ART networks, introduced by Carpenter and Grossberg [3,4], the learning process is not separated from its operation. Furthermore, ART neural networks are capable to add new states when necessary [8,9,12]. Therefore, this sort of ANNs were tested as a tool for classification of operational states in continuous monitoring of wind turbines.

The paper is organized in the following way. In section 2 the problem of classification of wind turbine states is discussed. The used ART networks - classical ART-2 network and fuzzy ART network are briefly described in section 3. The obtained results are presented in section 4.

2 Classification Problem of Wind Turbine States

In recent years large development of monitoring and diagnostic technologies for WTs has taken place. The growing number of installed systems created the need for analysis of gigabytes of data created every day by these systems. Apart from the development of several advanced diagnostic methods for this type of machinery there is a need for a group of methods, which will act as an "early warning". The idea of this approach could be based on a data driven algorithm, which would decide on a similarity of current data to the data, which are already known. In other words, the data from the turbine should be accounted for one of known states. If this is a state describing a failure, the human expert should be alarmed. If this is an unknown state, the expert should be informed about the situation and asked for a definition of such a new state.

This approach could be called "the blunt expert", which maybe sounds strange, but gives the most important feature of the proposed method. This approach may brake the biggest barrier of application of artificial neural networks (ANNs) in diagnostics, which is availability of significant amount of training data. As in real cases it is not possible to acquire it, it is only possible to train ANNs for a few cases covered by available data.

The problem of classification was investigated by several authors. One of the first works was research by Shuhui et al. [11], who compared classification techniques for the wind curve estimation. This work was often referenced by others, but from the ANN point of view it only multilayered feed-forward types of networks. Another important contribution was given by Kim [6], who compared performance of several classification methods. His experiments showed that unless the number of independent variables in the system is low, ANNs perform better than other methods. Again, the investigated network was the multi-layer feed-forward network trained by the back-propagation algorithm.

There are no works, known to us, which would consider application of ART networks for the classification of WT states. There were also works applying wind turbines for wind and power generation prediction, but this issue is outside the scope of this paper.

As the ART networks are capable to perform efficient classification and to recognize new states when necessary [3,4], we performed research of initial classification task. The goal of the experiment was verification of ART classification capabilities with comparison to the human expert. This type of data is acquired in the majority of cases and the successful classifier should create a reasonable number of classes, similar to these by a human expert. This task is the main goal of the following paper.

As such a classification was shown, it is thus possible to filter out states, which are known to be correct. The expert can then focus only on ”suspicious” states returned by the algorithm.

3 Characteristics of the Applied Neural Networks

3.1 ART-2 Network

Let us briefly recall ART-2 neural network properties tracing [10].

The ART-2 is an unsupervised neural network with based on adaptive resonance theory (ART). A typical ART-2 architecture, introduced in Carpenter and Grossberg (1987), is presented in Fig.1 (only one unit of each type is shown here). In the attentional sub-system, an input pattern s is first presented to the F_1 layer, which consists of six kinds of units - the W, X, U, V, P and Q cells. It then undergoes a process of activation, including normalization, noise suppression and updating. This results in an output pattern p from the F_1 layer. Responding to this output pattern, an activation is produced across F_2 layer through bottom-up weights b_{ij}. As the F_2 layer is a competitive layer with a winner-takes-all mode, only one stored pattern is a winner. It also represents the best matching pattern for the input pattern at the F_1 layer. Furthermore, the pattern of activation on the F_2 layer brings about an output pattern that is sent back to the F_1 layer via top-down weights t_{ji}. For the orienting sub-system, it contains a reset mechanism R and a vigilance parameter q to check for the similarity between the output pattern from the F_2 layer and the original input pattern from the F_1 layer. If both patterns are concordant, the neural network enters a resonant state where the adaptation of the stored pattern is conducted.

Otherwise, the neural network will assign an uncommitted (inhibitory) node on the F_2 layer for this input pattern, and thereafter, learn and transform it into a new stored pattern.

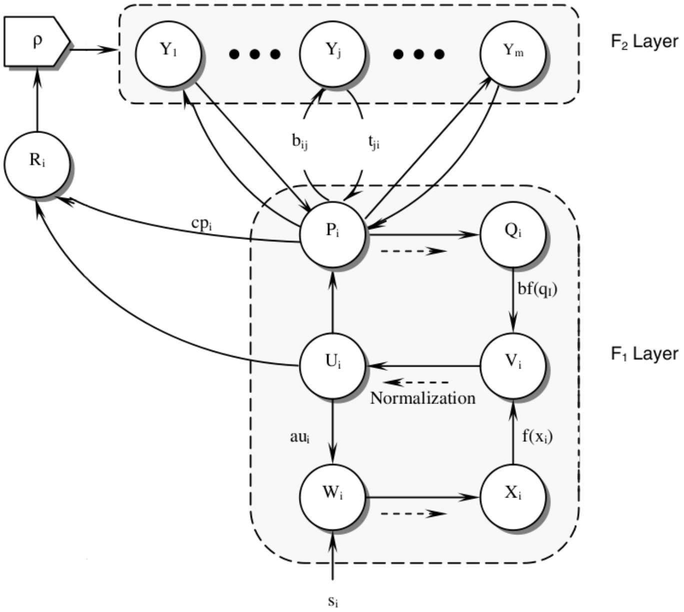


Fig. 1. ART-2 architecture

3.2 Fuzzy ART Network

The organization of a fuzzy ART network, introduced by Carpenter et all. [5], is presented in Fig.2. In comparison with a classical ART-2 network, the fuzzy ART network has an additional layer F_0 which transforms input vectors using so called complement coding.

The fuzzy ART network has a single weights matrix Z, which processes signals being sent both from F_1 to F_2 layer and vice versa. Operations done by the network are based on the fuzzy logic. The operator *fuzzy AND* is used for two vectors comparison - see [7].

Signal processing in fuzzy ART network is similar to processing in ART-2 network. The input signal vector, say X, is transformed by F_0 and F_1 layers producing signals T_j which are put to inputs of the F_2 layer. For the neuron which is excited most strongly a vigilance parameter q is used to check the similarity between the output pattern from the F_2 layer and the input pattern. If $p < \rho$ then the winner neuron is inhibited and other neuron in F_2 layer is searched. Otherwise, the weights matrix Z is modified in order to store the recognized vector features. The learning process is continued until the values of the matrix Z are stabilized.

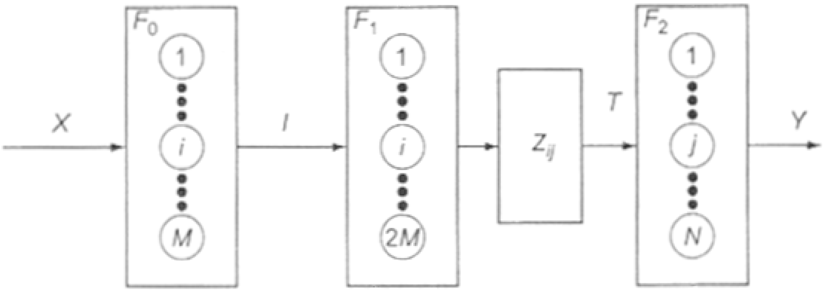


Fig. 2. Fuzzy ART architecture

4 Results

The practical case study was performed on data from one of wind turbines. The data covering the period from 11.09.2009 till 30.09.2009 were recorded every 10 minutes by the online monitoring system. The recorded data were current values and were not averaged. The data set included 2869 measurements. As the main goal of the work was to test applicability of ART-type ANNs for classification, in the first step we tried to use the network to achieve results similar to a human expert. Other training was not possible, as this type of networks performs only unsupervised learning.

The data set contained the most fundamental values, deciding about the operational state of the machine. These were: wind speed, rotational speed of the generator and the generated power. They are related, but only to some extent

Table 1. A number of classes recognized by ART-2 and fuzzy ART networks in dependence on the vigilance parameter. Used abbreviations: PW - Power and Wind (2D model), SW - Speed and Wind (2D model), PSW - Power, Speed and Wind (3D model).

Vigilance	ART-2 network			Fuzzy ART network		
parameter	PW	SW	PSW	PW	SW	PSW
0.4	1	1	1	1	2	2
0.5	1	1	1	2	4	2
0.6	1	1	1	2	4	4
0.65	1	1	1	2	5	5
0.7	1	1	1	3	7	5
0.75	1	1	1	4	8	9
0.8	1	1	1	6	10	11
0.9	1	2	2	14	24	16
0.9497	2	2	4	-	-	-
0.95	2	2	5	-	-	-
0.97	3	2	8	-	-	-
0.98	3	5	9	-	-	-
0,99	4	8	10	-	-	-

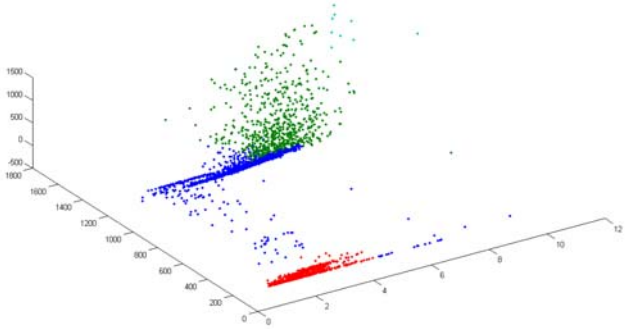


Fig. 3. Results for fuzzy ART, 3D model, vigilance parameter is equal to 0.55. According to a human expert's opinion 20 measurements were classified incorrectly which makes an error equal to $\frac{20}{2869} = 0.70\%$.

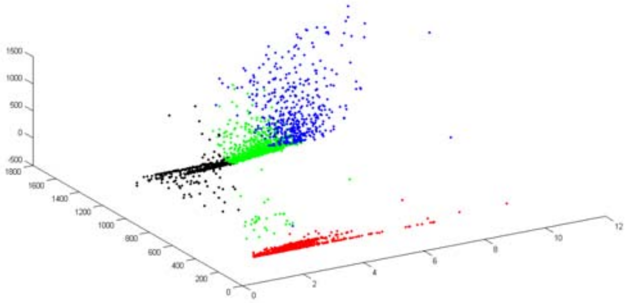


Fig. 4. Results for ART-2, 3D model, vigilance parameter is equal to 0.9497. According to a human expert's opinion 27 measurements were classified incorrectly which makes an error equal to $\frac{27}{2869} = 0.94\%$.

and in fact they are all independent variables. The selection of variables is the same as the human expert would use. Typically the operation of the WT can be divided in a few distinct states: stopped, transient between 0 and 1000 rpm, idle load (rpm about 1000 rpm, no load), low power, high power. Sometimes it is not necessary to distinguish all of them and the first two pairs are sometimes regarded as only two states (i.e. "stopped or transient" and "low power" including also idle mode). Very important advantage of chosen set is that it has only 3 variables and can be presented in a graphical way. Thus, it can be easily understood and compared with a human expert. Results create the basis and give some intuition for more advanced research.

The main idea of the research was to apply recorded data to the ART network and to investigate what is its behavior (i.e. how many states will be created) and how does this depends on the network parameters. The research was performed for two types of networks: ART-2 and fuzzy ART. It was found that the single most important parameter is the vigilance. Table 1 presents how the number of created states depends on this parameter. It can be seen that for the ART-2 network reasonable results are obtained for vigilance in the vicinity of 0.95. For fuzzy ART the optimum vigilance is around 0.6.

Apart from the number of states it was very important to check how the networks actually divide the data into states. It can be seen that both networks gave very interesting results, creating regions very similar to "natural" ones. In a few cases created regions stretched across a larger area than a human would create. This was most probably result of normalization and should be further investigated.

There are sight differences between the networks, but at this stage one can say that both are performing well and could be the basis for further research.

5 Concluding Remarks

Presented results belong to a broader research activity, aimed at automatic monitoring of rotating machinery. We are interested in investigation of several approaches, which can be applied in the engineering practice. Thus, one has to assume that learning sets are not available or cover only a part of machine states.The problem becomes much more the classification of the current state to one of previously known states or detection of a new state. Ideally, such a new state should be included for further classifications.

The initial attempt to the problem was investigation of PNN networks [1]. The study revealed classification capabilities much better than MLP, but handling of new states required retraining of the network.

The approach presented in this paper is much more interesting. To start with simple problems, we showed that ART-2 networks are capable to classify typical states of a wind turbine with efficacy very close to the human expert. Such a result is, in our opinion, very interesting and encouraging for further research. Further works will be conducted in two directions. Firstly, the dimensionality of the problem will be increased (i.e. more measurement channels will be taken into account). Secondly, the data with faults will be used to train the network.

References

1. Barszcz, T., Bielecki, A., Romaniuk, T.: Application of probabilistic neural networks for detection of mechanical faults in electric motors. Electrical Review 8/2009, 37–41 (2009)
2. Barszcz, T., Randall, R.B.: Application of spectral kurtosis for detection of a tooth crack in the planetary gear of a wind turbine. Mechanical Systems and Signal Processing 23, 1352–1365 (2009)
3. Carpenter, G.A., Grossberg, S.: A massively parallel architecture for a self-organizing neural pattern recognition machine. Computer Vision, Graphics, and Image Processing 37, 54–115 (1987)

4. Carpenter, G.A., Grossberg, S.: ART2: self-organization of stable category recognition codes for analog input pattern. Applied Optics 26, 4919–4930 (1987)
5. Carpenter, G.A., Grossberg, S., Rosen, D.B.: Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. Neural Networks 4, 759–771 (1991)
6. Kim, Y.S.: Performance evaluation for classification methods: A comparative simulation study. Expert Systems with Applications 37, 2292–2306 (2010)
7. Knosala, R.: Artificial Intelligence Methods Applications in Manufacturing Engineering. WNT, Warsaw (2002) (in Polish)
8. Korbicz, J., Obuchowicz, A., Uciński, D.: Artificial Neural Networks. Foundations and Applications. Academic Press PLJ, Warsaw (1994) (in Polish)
9. Rutkowski, L.: Neural networks and Neurocomputers. Technical University in Częstochowa Press, Częstochowa (1996) (in Polish)
10. Shieh, M.D., Yan, W., Chen, C.H.: Soliciting customer requirements for product redesign based on picture sorts and ART2 neural network. Expert Systems with Applications 34, 194–204 (2008)
11. Shuhui, L., Wunsch, D.C., O'Hair, E., Giesselmann, M.G.: Comparative analysis of regression and artificial neural network models for wind turbine power curve estimation. Journal of Solar Energy Engineering 123, 327–332 (2001)
12. Tadeusiewicz, R.: Neural Networks. Academic Press, Warsaw (1993) (in Polish)

Parallel Realisation of the Recurrent Elman Neural Network Learning

Jarosław Bilski and Jacek Smoląg

Department of Computer Engineering, Częstochowa University of Technology,
Częstochowa, Poland
{Jaroslaw.Bilski,Jacek.Smolag}@kik.pcz.pl

Abstract. The aim of this paper is to present a parallel architecture of Elman Recurrent Network learning algorithm. The solution is based on the high parallel cuboid structure to speed up computation. Parallel neural network structures are explicitly presented and the performance discussion is included.

1 Introduction

The Elman network proposed in 1990 by Jeffrey L. Elman [3] is an example of recurrent neural networks. The gradient method [8] is used to train the network. Dynamical neural networks have been researched by many scientists for the last decade [4], [5] . Typically neural networks learning algorithms are implemented on serial computer. They are required to use high computational load. Therefore high performance parallel structure was discussed in several papers, eg. [1], [2], [6], [7]. This paper presents a new concept of the parallel realisation of the Elman learning algorithm. The high performance of this new architecture is very promising for Elman neural networks. A single iteration of the parallel architecture requires much less computation cycles than a serial implementation. The final part of this work explains efficiency of the proposed architecture. The structure of the Elman network is shown in Fig. 1.

The Elman network contains N inputs and K neurons in hidden layer and M neurons in the network output. Moreover, all signals from outputs of neurons are connected as network inputs through unit time delay z^{-1}. Therefore the network input vector

$$\left[1, x_{1}^{(1)}(t), ..., x_{N}^{(1)}(t), x_{N+1}^{(1)}(t), ..., x_{N+K}^{(1)}(t)\right]^{T} \tag{1}$$

in the Elman network takes the form

$$\left[1, x_{1}^{(1)}(t), ..., x_{N}^{(1)}(t), y_{1}^{(1)}(t-1), ..., y_{K}^{(1)}(t-1)\right]^{T} \tag{2}$$

In the recall phase the network is described by

$$s_{k}^{(1)} = \sum_{i=0}^{N+K} w_{ki}^{(1)} x_{i}^{(1)} \tag{3}$$

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 19–25, 2010.
© Springer-Verlag Berlin Heidelberg 2010

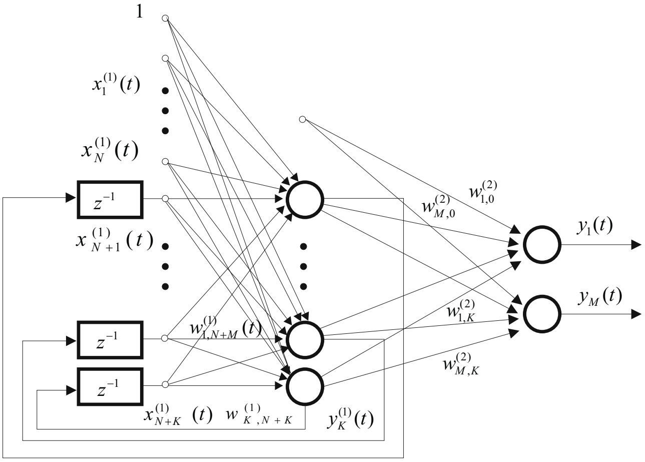


Fig. 1. Structure of the Elman network

$$y_k^{(1)}(t) = f_1(s_k^{(1)}(t)) \tag{4}$$

We denote by $\mathbf{w}_k^{(1)} = [w_{k0}^{(1)}, ...w_{k,N+K}^{(1)}]$ vector of weights of the $k-th$ neuron and $\mathbf{W}^{(1)} = [\mathbf{w}_1^{(1)}, ..., \mathbf{w}_K^{(1)}]$. In the second layer:

$$s_j^{(2)} = \sum_{k=0}^{K} w_{jk}^{(2)} x_k^{(2)} = \sum_{k=0}^{K} w_{jk}^{(2)} y_k^{(1)} \tag{5}$$

$$y_j^{(2)}(t) = f_2(s_j^{(2)}(t)) \tag{6}$$

To learn the Elman neural network we minimise the following criterion

$$J\left(t\right) = \tfrac{1}{2} \sum\nolimits_{j=1}^{M} \varepsilon_j^{(2)^2}\left(t\right) = \tfrac{1}{2} \sum\nolimits_{j=1}^{M} \left(d_j^{(2)}\left(t\right) - y_j^{(2)}\left(t\right)\right)^2 \tag{7}$$

where $\varepsilon_j^{(2)}(t)$ is defined as

$$\varepsilon_j^{(2)}(t) = d_j^{(2)}(t) - y_j^{(2)}(t). \tag{8}$$

Weights in the l-th layer are updated by

$$w_{\alpha\beta}^{(l)}(t+1) = w_{\alpha\beta}^{(l)}(t) - \eta \nabla_{\alpha\beta}^{(l)} J\left(t\right) \tag{9}$$

where $w_{\alpha\beta}^{(l)}$ is $\beta - th$ weight for $\alpha - th$ neuron in $l - th$ layer and

$$\nabla_{\alpha\beta}^{(l)} J\left(t\right) = \frac{dJ(t)}{w_{\alpha\beta}^{(l)}(t)} = \frac{d\frac{1}{2}\sum_{j=1}^{M} \varepsilon_j^{(2)^2}\left(t\right)}{dw_{\alpha\beta}^{(l)}} = -\sum\nolimits_{j=1}^{M} \varepsilon_j^{(2)}\left(t\right) \frac{dy_2\left(t\right)}{dw_{\alpha\beta}^{(l)}} \tag{10}$$

By minimising equation (7) for the second layer weights we obtain

$$\begin{aligned}
&\nabla^{(2)}_{\alpha\beta} J\left(t\right) = -\sum\nolimits_{j=1}^{M} \varepsilon^{(2)}_j\left(t\right) \frac{df_2\left(s^{(2)}_j(t)\right)}{dw^{(2)}_{\alpha\beta}} = \\
&= -\sum\nolimits_{j=1}^{M} \varepsilon^{(2)}_j\left(t\right) \frac{df_2\left(s^{(2)}_j(t)\right)}{ds^{(2)}_j(t)} \sum\nolimits_{k=0}^{K} \frac{dw^{(2)}_{jk}(t)}{dw^{(2)}_{\alpha\beta}} y^{(1)}_k\left(t\right) = \\
&= -\varepsilon^{(2)}_\alpha\left(t\right) \frac{df_2\left(s^{(2)}_\alpha(t)\right)}{ds^{(2)}_\alpha(t)} y^{(1)}_\beta\left(t\right) = -\varepsilon^{(2)}_\alpha\left(t\right) f^{'}_{2,\alpha}\left(t\right) y^{(1)}_\beta\left(t\right)
\end{aligned} \tag{11}$$

By minimising equation (7) for the first layer weights we obtain

$$\begin{aligned}
&\nabla^{(1)}_{\alpha\beta} J\left(t\right) = -\sum\nolimits_{j=1}^{M} \varepsilon^{(2)}_j\left(t\right) \frac{df_2\left(s^{(2)}_j(t)\right)}{ds^{(2)}_j(t)} \sum\nolimits_{k=1}^{K} \frac{dy^{(1)}_k(t)}{dw^{(1)}_{\alpha\beta}} w^{(2)}_{jk}\left(t\right) = \\
&= -\sum\nolimits_{k=1}^{K} \sum\nolimits_{j=1}^{M} \varepsilon^{(2)}_j\left(t\right) \frac{df_2\left(s^{(2)}_j(t)\right)}{ds^{(2)}_j(t)} \frac{dy^{(1)}_k(t)}{dw^{(1)}_{\alpha\beta}} w^{(2)}_{jk}\left(t\right) = \\
&= -\sum\nolimits_{k=1}^{K} \frac{dy^{(1)}_k(t)}{dw^{(1)}_{\alpha\beta}} \sum\nolimits_{j=1}^{M} \varepsilon^{(2)}_j\left(t\right) \frac{df_2\left(s^{(2)}_j(t)\right)}{ds^{(2)}_j(t)} w^{(2)}_{jk}\left(t\right)
\end{aligned} \tag{12}$$

By defining

$$\varepsilon^{(1)}_k \overset{def}{=} \sum\nolimits_{j=1}^{M} \varepsilon^{(2)}_j\left(t\right) \frac{df_2\left(s^{(2)}_j\left(t\right)\right)}{ds^{(2)}_j\left(t\right)} w^{(2)}_{jk}\left(t\right) = \sum\nolimits_{j=1}^{M} \varepsilon^{(2)}_j\left(t\right) f^{'}_{2,j}\left(t\right) w^{(2)}_{jk}\left(t\right) \tag{13}$$

we obtain

$$\nabla^{(1)}_{\alpha\beta} J\left(t\right) = -\sum\nolimits_{k=1}^{K} \frac{dy^{(1)}_k\left(t\right)}{dw^{(1)}_{\alpha\beta}} \varepsilon^{(1)}_k\left(t\right) \tag{14}$$

The derivative

$$\frac{dy^{(1)}_k(t)}{dw^{(1)}_{\alpha\beta}} = f_1'\left(s^{(1)}_k(t)\right) \sum_{i=0}^{N+K} \frac{d(w^{(1)}_{ki} x^{(1)}_i(t))}{dw^{(1)}_{\alpha\beta}} \tag{15}$$

can be calculated

$$\begin{aligned}
&\frac{dy^{(1)}_k(t)}{dw^{(1)}_{\alpha\beta}} = f_1'\left(s^{(1)}_k(t)\right) \left(\delta_{k\alpha} x^{(1)}_\beta + \sum_{i=N+1}^{N+K} w^{(1)}_{ki} \frac{dx^{(1)}_i(t)}{dw^{(1)}_{\alpha\beta}}\right) = \\
&= f_1'\left(s^{(1)}_k(t)\right) \left(\delta_{k\alpha} x^{(1)}_\beta + \sum_{i=N+1}^{N+K} w^{(1)}_{ki} \frac{dy^{(1)}_{i-N}(t-1)}{dw^{(1)}_{\alpha\beta}}\right) = \\
&= f_1'\left(s^{(1)}_k(t)\right) \left(\delta_{k\alpha} x^{(1)}_\beta + \sum_{i=1}^{K} w^{(1)}_{k,i+N} \frac{dy^{(1)}_i(t-1)}{dw^{(1)}_{\alpha\beta}}\right)
\end{aligned} \tag{16}$$

and the weights can be updated as follows

$$w^{(1)}_{\alpha\beta}(t+1) = w^{(1)}_{\alpha\beta}(t) - \eta \sum_{i=1}^{K} \varepsilon^{(1)}_i(t) \frac{dy^{(1)}_i(t)}{dw^{(1)}_{\alpha\beta}} \tag{17}$$

2 Parallel Realisation

The second layer is the simple feedforward layer and can be trained by using equations (9) and (11). The structure realising training phase for this layer is presented in Fig. 2. It updates all weights in one step and produces error signals $\varepsilon_i^{(1)}(t)$ for first layer in M steps. The structure of single processing element is shown in Fig. 3.

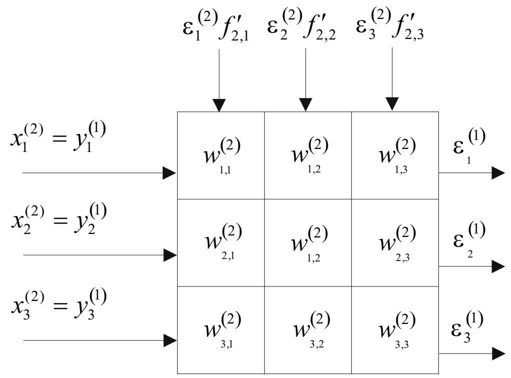


Fig. 2. The second layers weights structure

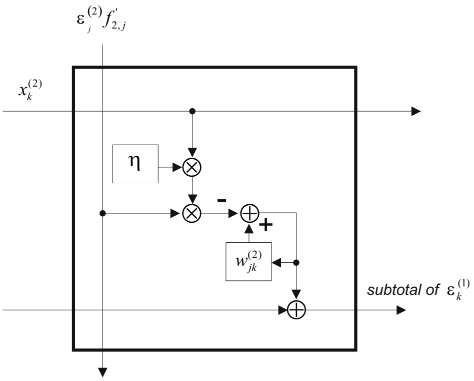

$\varepsilon_j^{(2)} f'_{2,j}$
$x_k^{(2)}$
η
-
+
$w_{jk}^{(2)}$
subtotal of $\varepsilon_k^{(1)}$

Fig. 3. Structure of processing elements for second layer

The main idea of the parallel realisation of the Elman learning algorithm is based on the cuboid computational matrix for the first layer (see Fig. 4). The matrix contains processor elements realising operations given in equations (16) and (17). The cuboid matrix dimensions are $K \times K \times (K + N + 1)$. To make it clear in all figures dimension K is set to 3. Any element of the cuboid matrix for indexes i and $\alpha\beta$ is obtained by multiplication of vector of cuboid matrix elements with index $\alpha\beta$ by the $i-th$ weight subvector (see Eqn. 16). At the same

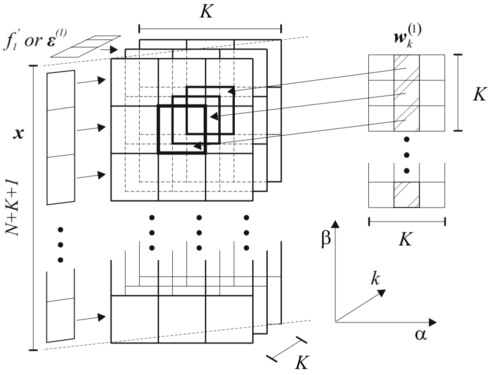


Fig. 4. Idea of the cuboid computation matrix

time the whole vector of cuboid elements for index $\alpha\beta$ can be multiplied by the weight submatrix $K \times K$. In Fig. 5 accurate data flow is presented. The weight submatrix is entered parallely to all elements of the cuboid matrix. The signals $\mathbf{f}'$ and $\mathbf{x}$ are simultaneously delivered. Figure 7 shows horizontal section of the cuboid matrix. Note that weights and first derivatives lead to all elements but input x_β leads only to elements on the main diagonal. The sum in equation (16) is computed in K steps and its subtotal circles between $i-th$ elements with index $\alpha\beta$. All elements are computed at the same time. Figure 6 shows the structure of

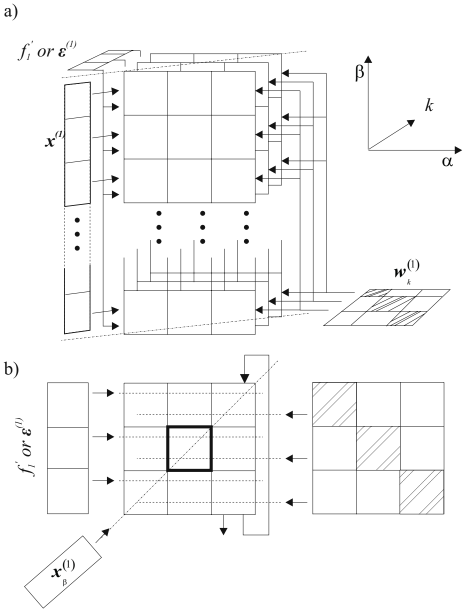


Fig. 5. Data flow in a) computational cuboid, b) one layer for $\beta = const$

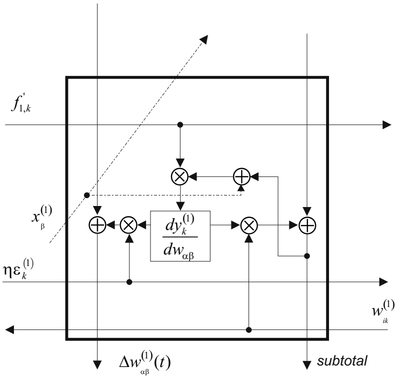

$f'_{1,k}$
$x_\beta^{(1)}$
$\eta\varepsilon_k^{(1)}$
$\frac{dy_k^{(1)}}{dw_{\alpha\beta}}$
$w_{ik}^{(1)}$
$\Delta w_{\alpha\beta}^{(1)}(t)$
subtotal

Fig. 6. Single computational element of cuboid matrix

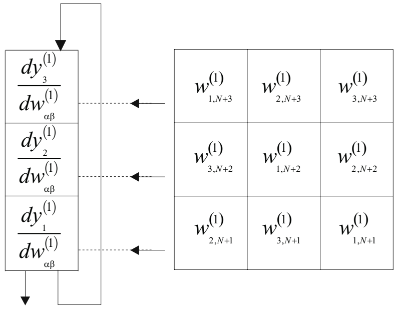


Fig. 7. Weight distribution in learning time

one processor element. It computes in K steps new derivative $\frac{d(y_i^{(1)}(t))}{dw_{\alpha\beta}^{(1)}}$ and in the next K steps values of weights' changes described by equation (16). The dotted lines appear only in the elements of the main diagonal. Figure 7 depicts the distribution of weights during learning phase. Note that columns of submatrix of $\mathbf{W}$ are placed diagonally. The elements are simultaneously enabled to compute.

3 Conclusion

In this paper the parallel realisation of the Elman neural network was presented. We assume that all multiplications and additions operations take the same time unit. We can compare computational performance of the Elman parallel implementation with sequential architectures up to 10 inputs (N) and 10 neurons (K) in the hidden layer of neural network. For simplicity the number of output neurons (M) is equal to the number of hidden neurons. Computational complexity

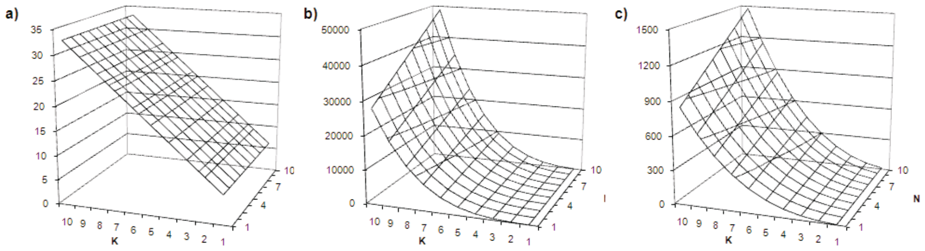


Fig. 8. Number of times cycles in a) classical, b) parallel implementation and c) performance factor parallel/classical

of Elman learning is of order $\mathcal{O}(K^4)$ and equals $2K^4 + K^3(2N+5) + K^2(3N+4) + 2K + 4KM + 2M$. In the proposed parallel solution each iteration requires only $2K + M + 3$ time units (see Fig. 8).

Performance factor of parallel realisation of the Elman algorithm achieves nearly 1500 for 10 inputs and 10 output neurons and it grows fast when the number of network inputs or outputs grows. We observed that the performance of the proposed solution is promising.

References

1. Bilski, J., Litwiński, S., Smoląg, J.: Parallel realisation of QR algorithm for neural networks learning. In: Rutkowski, L., Siekmann, J.H., Tadeusiewicz, R., Zadeh, L.A. (eds.) ICAISC 2004. LNCS (LNAI), vol. 3070, pp. 158–165. Springer, Heidelberg (2004)
2. Bilski, J., Smoląg, J.: Parallel realisation of the RTRN neural network learning. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 11–16. Springer, Heidelberg (2008)
3. Elman, J.L.: Finding structure in time. Cognitive Science 14, 179–211 (1990)
4. Kolen, J.F., Kremer, S.C.: A Field Guide to Dynamical Recurrent Neural Networks. IEEE Press, Los Alamitos (2001)
5. Korbicz, J., Patan, K., Obuchowicz, A.: Dynamic neural networks for process modelling in fault detection and isolation. Int. J. Appl. Math. Comput. Sci. 9(3), 519–546 (1999)
6. Smoląg, J., Bilski, J.: A systolic array for fast learning of neural networks. In: Proc. of V Conf. Neural Networks and Soft Computing, Zakopane, pp. 754–758 (2000)
7. Smoląg, J., Rutkowski, L., Bilski, J.: Systolic array for neural networks. In: Proc. of IV Conf. Neural Networks and Their Applications, Zakopane, pp. 487–497 (1999)
8. Williams, R., Zipser, D.: A learning algorithm for continually running fully recurrent neural network. Neural Computation, 270–280 (1989)

The Investigating of Influence of Quality Criteria Coefficients on Global Complex Models

Grzegorz Dralus

Rzeszow University of Technology,
Department of Electrical Engineering and Informatics Fundamentals
Pola 2 str, 35-959 Rzeszow, Poland
gregor@prz.edu.pl

Abstract. In the paper global modeling of complex systems with regard to quality of local models of simple plants are discussed. Complex systems consist of several sub-systems. As a global model multilayer feedforward neural networks were used. It is desirable to obtain an optimal global model, as well as optimal local models. A synthetic quality index as a sum of a global quality criterion and local quality indexes is defined. By optimization of the synthetic quality index the global model is obtained with regard to the quality of local models of sub-systems. The quality index of the global model contains coefficients which define the participation of the local criteria in the synthetic quality criterion. The investigation of influence of these coefficients on the quality of the global model of the complex static system is discussed. The investigation is examined by a complex system which consists of two nonlinear simple plants.

Keywords: complex system, neural networks, global model.

1 Introduction

The numerous problems of projecting of complex systems steering are related to suitable constructing the models of complex systems. The regard of the complex system as the whole model is one of principal problem of the global model study as well as the suitable quality assurance of approximation of individual elements of system.

The classic task of modeling the complex system depends on finding the optimum values of parameters of received mathematical model in support determined quality criterion. The mathematical methods of identification of complex systems depend on their decomposition into component parts. The next step is to build local models of particular parts of the complex system (simple plants) and search their optimal parameters. Then the complex model is composed of optimal local models. Such model is not globally optimal because, during the adjusting process of local models parameters, the interactions of component parts of complex system are not taken into account [1]. The neural networks which possess the ability to approximate each non-linear function allow to construct

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 26–33, 2010.
© Springer-Verlag Berlin Heidelberg 2010

and to settle of parameters of global models. In foundation a global model has to reflect the structure of a complex system during his work, and also to reflect the mutual interactions of component parts of complex system during adjusting of parameters of the global model. Generally, neural networks allow to construct more exact models than decomposition methods [4], [5].

2 Modeling of Static Complex Systems

The most known modeling method of complex systems depends on its decomposition on simple plants [1]. The simple plants can be modeled by arbitrary method as independent simple plants without regard of fact, that they are the parts of a complex system. After obtainment of the optimal parameters of simple models, the complex model is composed the same as the complex system. In this way a locally optimal complex model is built, but it is not globally optimal model. However, to get a globally optimal model multilayer neural networks can be applied. The neural networks permit to build a global model because an internal structure of the complex model is adequate to the complex system. The learning process of neural networks allows to achieve satisfactory parameters of a global model for which the quality of model is good sufficiently. Complex systems can have varied structure [1] ,[2], [6]. In the paper a complex system is cascade connected simple plants, which is often present in industrial plants.

2.1 Global Modeling

In global modeling global models have the same structure as complex systems and parameters of global models are sought. The complex system as simple plants connected in series (appointed as O_1 .. O_R) is shown in Figure 1. The complex model is divided into R simple models. In the complex model it is possible to indicate simple models denote as M_r (r=1,2,...,R), which correspond to the suitable simple plants. The output $\hat{\boldsymbol{y}}^{(r)}$ of the r-th simple model is the input to the next simple model (M_{r+1}) similarly how it is in the complex system. The simple models can be also local models. This fallows when the input of r-th local model is taken form the r-1 simple plant. Then output of r-th local model is denoted as $\bar{\boldsymbol{y}}^{(r)}$. Thus M_r is the r-th simple model as a part of global model but M_r is also the r-th local model.

On the basis of the difference between the output of the global model and the output complex system quality index of the global model is formulated the following:

$$Q(\boldsymbol{W}) = \frac{1}{2}\left(\hat{\boldsymbol{y}}^{(R)} - \boldsymbol{y}^{(R)}\right)^2 = \frac{1}{2}\sum_{k=1}^{K}\sum_{j=1}^{J_R}\left(\hat{y}_j^{(R),k} - y_j^{(R),k}\right)^2 . \tag{1}$$

where: $\boldsymbol{W}$ are the parameters (the weights) of the global model, K - the number of data elements of the output vector $\boldsymbol{y}^{(R)}$, J_R - the number of outputs signals in the output vector $\boldsymbol{y}^{(R)}$.

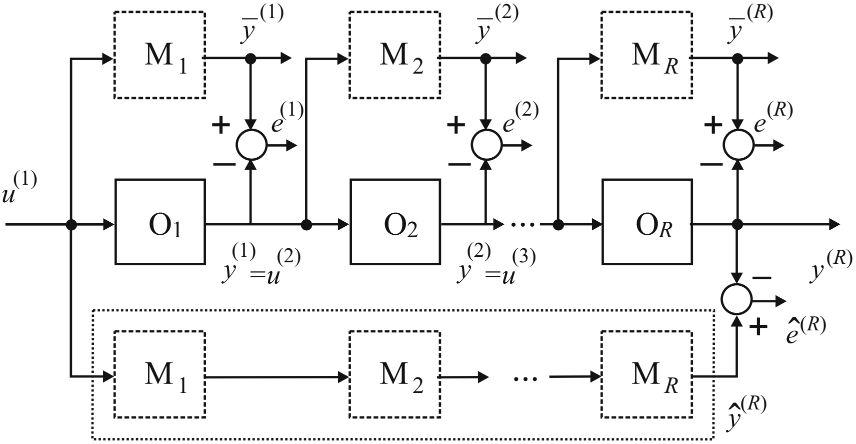


Fig. 1. Global and local models of a complex system

On the difference basis between the output r-th local model and the output suitable simple plant a quality index of the r-th local model is defined:

$$Q_r(\boldsymbol{w}^{(r)}) = \frac{1}{2}\left(\bar{\boldsymbol{y}}^{(r)} - \boldsymbol{y}^{(r)}\right)^2 = \frac{1}{2}\sum_{k=1}^{K}\sum_{j=1}^{J_r}\left(\bar{y}_j^{(r),k} - y_j^{(r),k}\right)^2 . \tag{2}$$

where: $\boldsymbol{w}^{(r)}$- the parameters (weights) the r-th simple (local) model, J_r - the number of outputs the r-th simple plant.

The performance index of the global model with regard to the quality of R local models is accept as weighed sum of the global quality indexes (1) and the local quality indexes (2):

$$Q_s(\boldsymbol{W}) = \alpha_0\, Q(\boldsymbol{W}) + \alpha_1\, Q_1(\boldsymbol{w}^{(1)}) + \cdots + \alpha_R\, Q_R(\boldsymbol{w}^{(R)}). \tag{3}$$

The performance index (3) is named the synthetic quality index and it is shown in detail form as:

$$Q_s(\boldsymbol{W}) = \frac{1}{2}\,\alpha_0 \sum_{k=1}^{K}\sum_{j=1}^{J_R}\left(\hat{y}_j^{(R),k} - y_j^{(R),k}\right)^2 + \frac{1}{2}\sum_{r=1}^{R}\alpha_r \sum_{k=1}^{K}\sum_{j=1}^{J_r}\left(\bar{y}_j^{(r),k} - y_j^{(r),k}\right)^2 . \tag{4}$$

where: $\boldsymbol{W}$=[$\boldsymbol{w}^{(1)}$ $\boldsymbol{w}^{(2)}$ ... $\boldsymbol{w}^{(R)}$] is the matrix of parameters (weights) of the global model, $\alpha = [\alpha_0\ \alpha_1\ \ldots \alpha_R]$ - weighted coefficients, $0 < \alpha_r < 1$ and $\sum_{r=1}^{R}\alpha_r = 1$.

The synthetic quality criterion (4) allow us to achieve a compromise between a global optimal model of a complex system and local optimal models of subsystems (simple plants). The minimization of the Q_s index leads to obtaining the global model with regard to local models. The coefficient α_0 defines the influence of the global quality criterion (1) on the index Q_s. The coefficients α define the influence of the local quality criteria (2) on the synthetic quality criterion Q_s. By appropriate choice of the α values we can point out to the meaning of local models in a complex system model.

2.2 Learning Algorithm

The quality index Q_s is the sum of squares of errors of local models and the global model. It is differentiable and has continuous derivative, which is necessary condition in calculation of gradient algorithms. In order to minimise the error function $Q_s(\boldsymbol{W})$, the gradient descend method is applied. In order to learn neural network the following weight increment is calculated:

$$\triangle\, w_{ji} = -\eta \frac{\partial Q_s(\boldsymbol{W})}{\partial w_{ji}}\ . \tag{5}$$

The changes of weights in the output layer of net after gradient calculations of the formula (5) are computed as the following:

$$\begin{aligned}\triangle\, w_{ji}^{(R),M} = & -\eta \sum_{k=1}^{K} f'(z_j^{M,k}) \left(\alpha_0 \left(\hat{y}_j^{(R),k} - y_j^{(R),k}\right) + \alpha_R \left(\bar{y}_j^{(R),k} - y_j^{(R),k}\right)\right) \\ & \times u_i^{M-1,k} .\end{aligned} \tag{6}$$

in the hidden layers (where $z_j^{m,k} = \sum_{i=0}^{Im-1} w_{ji}^{m} u_i^{m-1,k}$)

$$\triangle\, w_{ji}^{(r),m} = -\eta \sum_{k=1}^{K} f'(z_j^{m,k}) \left(\sum_{l=1}^{L_{m+1}} \delta_l^{(r),m+1,k} w_{lj}^{(r),m+1} \right) u_i^{m-1,k}. \tag{7}$$

in the 'binding' hidden layers (the notion 'binding' is explained in section 3)

$$\begin{aligned}\triangle\, w_{ji}^{(r),m} = & -\eta \sum_{k=1}^{K} f'(z_j^{m,k}) \times \\ & \left(\sum_{l=1}^{L_{m+1}} \delta_l^{(r+1),m+1,k} w_{lj}^{(r+1),m+1} + \alpha_r \left(\bar{y}_j^{(r),k} - y_j^{(r),k}\right) \right) u_i^{m-1,k} .\end{aligned} \tag{8}$$

Formulas (6), (7) and (8) are the complex gradient backpropagation learning algorithm for the global model with regard to quality local models. Moreover, a faster learning algorithm called the complex Rprop was also developed [3], [7].

3 Simulations

The complex system used to simulations is shown in Figure 2. It consists of two nonlinear simple plants connected in series. The input signal vector is appointed as $\boldsymbol{u}$. The signal $\boldsymbol{v}$ is the output of the first simple plant and it is the input to

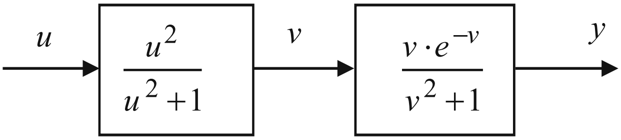


Fig. 2. The complex system as two connected nonlinear simple plants

the second simple plant. The signal $\boldsymbol{y}$ is the output of the second simple plant and it is the external output of the complex system.

The global model for the complex system in Figure 2 was accepted as the six layers' neural network shown in Figure 3. It has following structure: 1-4T-2T-1L-4T-2T-1L (6 layers). In this model it can be distinguished two simple models as 4T-2T-1L, connected in series, which are suitable with simple objects. The simple models have two hidden layers with nonlinear functions of activation (hyperb. tangent (T)), and the linear (L) activation functions in the output layers. The 'binding' hidden layer is the layer which connects (binds) simple models in a complex model. The output of this layer corresponds to the output of suitable simple plant. In this model the 'binding' hidden layer is the third layer (as 1L).

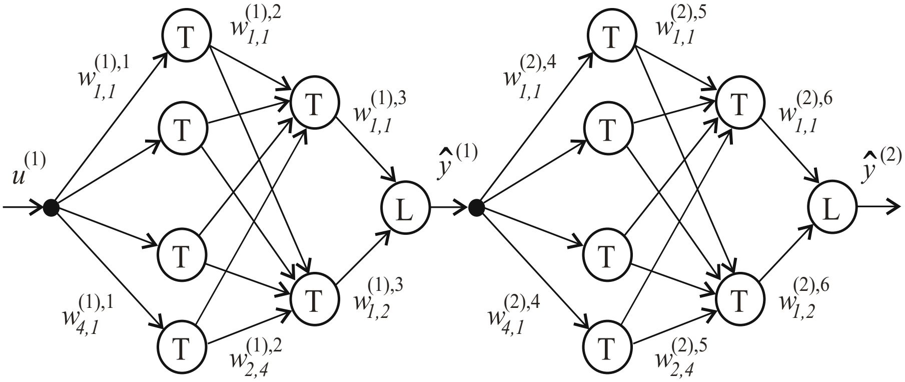


Fig. 3. The multilayer neural network as the global complex model

Random generated teaching vector contains 101 points of data from the range 0..5. Testing data contains 101 points spread in range 0 to 5 with step 0.05. The initial weights e.g. the initial parameters of models were generated according to Nguyen−Widrow rule [3]. For the complex gradient algorithm the learning rate $\eta = 0.005$ was accepted. The learning process after 10000 epochs was stopped.

The values of local quality criteria Q_1 and Q_2, the global criterion Q and the synthetic quality criterion Q_s for the testing data for coefficients $\alpha_0 = 1$ and $\alpha_0 = 0.1$ are shown in Table 1 and for $\alpha_0 = 0.5$ in Table 2. The change of coefficients every time carried out $\alpha_1 + \alpha_2 = 1$. Values of the quality indexes for the complex gradient learning algorithm are shown in Figure 4 and Figure 5. In

Table 1. Investigation performance indexes for testing data for $\alpha_0 = 1$ and $\alpha_0 = 0.1$

	$\alpha_0 = 1$				$\alpha_0 = 0.1$			
α_1	Q_1	Q_2	Q	Q_s	Q_1	Q_2	Q	Q_s
0.001	15.700	0.790000	0.720000	1.5249	7.05000	0.116000	0.12300	0.2459
0.01	14.000	1.540000	1.370000	3.0346	1.61600	0.001850	0.08670	0.1047
0.1	0.1460	0.001170	0.005870	0.0215	0.18100	0.000165	0.01550	0.0337
0.2	0.0978	0.001520	0.003370	0.0241	0.10200	0.000167	0.00514	0.0257
0.4	0.0443	0.000553	0.002060	0.0201	0.04470	0.000183	0.00420	0.0222
0.5	0.0313	0.000241	0.001670	0.0174	0.03310	0.000225	0.00351	0.0202
0.6	0.0223	0.000095	0.001270	0.0147	0.02530	0.000312	0.00294	0.0182
0.8	0.0143	0.000049	0.000753	0.0122	0.01610	0.000788	0.00280	0.0158
0.9	0.0119	0.000063	0.000620	0.0113	0.01330	0.000134	0.00333	0.0153
0.99	0.0106	0.000079	0.000569	0.0111	0.01170	0.000256	0.00478	0.0164
0.999	0.0105	0.000081	0.000586	0.0111	0.01160	0.000303	0.00529	0.0169

Table 2. Investigation performance indexes for testing data for $\alpha_0 = 0.5$

	$\alpha_0 = 0.5$			
α_1	Q_1	Q_2	Q	Q_s
0.001	14.7000	1.790000	1.72000	3.5229
0.01	1.47200	0.006870	0.03650	0.0580
0.1	0.15900	0.001140	0.00767	0.0246
0.2	0.09820	0.000878	0.00362	0.0240
0.4	0.04390	0.000339	0.00282	0.0206
0.5	0.03180	0.000212	0.00231	0.0183
0.6	0.02400	0.000151	0.00184	0.0163
0.8	0.01560	0.000145	0.00126	0.0138
0.9	0.01260	0.000178	0.00114	0.0125
0.99	0.01110	0.000248	0.00118	0.0122
0.999	0.01100	0.000258	0.00119	0.0122

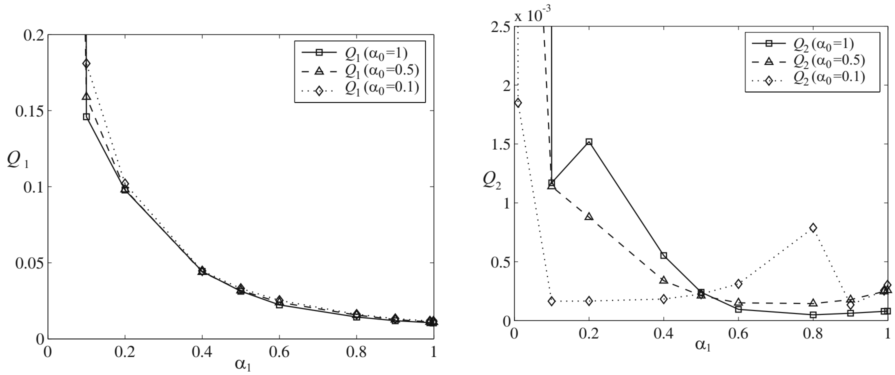


Fig. 4. Values of Q_1 and Q_2 indexes after 10000 epochs for testing data

Figure 4 is shown the influence of the coefficients α_0 on the local quality criteria Q_1 and Q_2. In Figure 5 is shown the influence the α_0 coefficients on the global quality index Q and the synthetic index Q_s.

The synthetic quality index Q_s directly depends on all α coefficients. However, α coefficients indirectly impact by the model's parameters $\boldsymbol{W}$ on local indexes Q_1, Q_2 and the global index Q. The increase of the value coefficient α_1 (the same the fall α_2) produces the monotonic fall of quality index Q_1. The smallest value of index Q_1 is achieved for the maximum value of coefficient α_1 which is equal 0.999 (see Figure 4)). When the coefficient α_2 grows the value of index Q_2 has minimum which depends on value of α_0 (see Figure 4)).

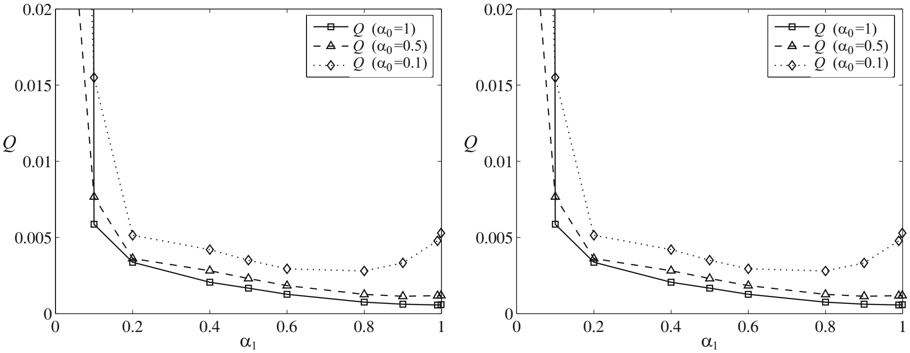


Fig. 5. Values of Q and Q_s indexes after 10000 epochs for testing data

Small values of Q index can be achieved for high values of α_1 coefficient (see Figure 5). The synthetic quality index Q_s also falls when coefficient α_1 is growing. The increase of α_0 coefficient leads to lower values of Q and Q_s indexes.

The best local models can be obtained in other points than the global model especially for high α_1. But the best global model with regard to quality of local models can be achieved especially around half range of α_1 coefficient. During the learning process the local models affect on the synthetic quality index Q_s and the global quality index Q. The learning process can be stopped when chosen quality index achieves a determined small value. This makes possible the tuning of the global model parameters until achievement of satisfactory exactitude of the model with regard to quality of local models. The learning process of the network can be also finished if the global quality index Q reaches a determined low value.

4 Conclusions

The global model of the static complex system with with taking into account the quality of local models was discussed. As the models the multilayer neural networks were introduced. The multi-layer neural networks are useful tool for complex systems modeling. The influence of the coefficients on the local models quality and the global model quality was investigated. Results of the investigation were shown for the testing data. Obtained results show that, by suitable selection of α coefficients we have the influence on the quality of local models and the quality of the global model. Because now we know the course of changes of local and global quality indexes so we can choose the values of α coefficients to obtain an optimal global model with regard to quality of local models. The good quality of local models fosters the good quality of a global model.

References

1. Bubnicki, Z.: Identification of Control Plants, Oxford, New York. Elsevier, Amsterdam (1980)
2. Dahleh, M.A., Venkatesh, S.R.: System Identification of Complex Systems. IEEE Proceedings of Problem Formulation and Results 3, 2441–2446 (1997)
3. Dralus, G., Swiatek, J.: A modified backpropagation algorithm for modelling static complex systems using neural network. In: Proceedings of 5th International Conference: Neural Network and Soft Computing, Zakopane, pp. 463–468 (2000)
4. Dralus, G., Swiatek, J.: Static neural network in global modelling of complex systems. In: The 14th International Conference on Systems Engineering, Coventry, pp. 547–551 (2000)
5. Dralus, G., Swiatek, J.: Global network modeling of complex systems with respect of local models quality. In: Proceedings of Fifteenth International Conference on System Engineering ICSE 2002, August 6-8, 2002, pp. 218–226. Univ. of Nevada, Las Vegas (2002)
6. Jozefczyk, J.: Decision making problems in complex of operations systems. Wroclaw University of Technology Press, Wroclaw (2002) (in Polish)
7. Riedmiller, M., Braun, H.: RPROP – a fast adaptive learning algorithm. Technical Report, University Karlsruhe (1992)

Quasi-parametric Recovery of Hammerstein System Nonlinearity by Smart Model Selection

Zygmunt Hasiewicz, Grzegorz Mzyk, and Przemysław Śliwiński

Institute of Computer Engineering, Control and Robotics
Wrocław University of Technology
Janiszewskiego 11/17, 50-372 Wrocław, Poland
grzegorz.mzyk@pwr.wroc.pl

Abstract. In the paper we recover a Hammerstein system nonlinearity. Hammerstein systems, incorporating nonlinearity and dynamics, play an important role in various applications, and effective algorithms determining their characteristics are not only of theoretical but also of practical interest. The proposed algorithm is quasi-parametric, that is, there are several parametric model candidates and we assume that the target nonlinearity belongs to the one of the classes represented by the models. The algorithm has two stages. In the first, the neural network is used to recursively filter (estimate) the nonlinearity from the noisy measurements. The network serves as a teacher/trainer for the model candidates, and the appropriate model is selected in a simple tournament-like routine. The main advantage of the algorithm over a traditional one stage approach (in which models are determined directly from measurements), is its small computational overhead (as computational complexity and memory occupation are both greatly reduced).

Keywords: system identification, structure detection, Hammerstein system, wavelet neural network.

1 Introduction

1.1 Types of Knowledge: Classification of Approaches

In the paper we propose the cooperation between parametric and nonparametric methods for system modeling. The term 'parametric' means that estimated nonlinearity can be described with the use of finite number of parameters. The nonparametric approach is applied for smart selection of the best parametric model of nonlinear characteristic from a given finite class of models. First, nonparametric estimates are continuously (recursively) computed from the learning pairs on the grid of N_0 input points and support selection of one from competing models. The parameters of the best model in the selected class are then obtained by the nonlinear least squares, and broad variety of optimization algorithms (also soft methods, e.g. genetic, tabu search, simulated annealing, particle swarming) can be applied in this stage, depending on the specifics of the optimization criterion. If the resulting parametric approximation is not satisfying, and the number

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 34–41, 2010.
© Springer-Verlag Berlin Heidelberg 2010

of measurements is large enough, the residuum between the system output and the model output is used for its nonparametric refinement.

2 Statement of the Problem

2.1 Class of Systems

The Hammerstein system is built of a static non-linearity, $\mu()$, and a linear dynamics, with the impulse response $\{\gamma_i\}_{i=0}^{\infty}$, connected in a cascade and described by the following set of equations: $y_k = v_k + z_k$, $v_k = \sum_{i=0}^{\infty} \gamma_i w_{k-i}$, $w_k = \mu(u_k)$, or equivalently

$$y_k = \sum_{i=0}^{\infty} \gamma_i \mu(u_{k-i}) + z_k, \tag{1}$$

where u_k and y_k denote the system input and output at time k, respectively, and z_k is the output noise (see Fig.1)

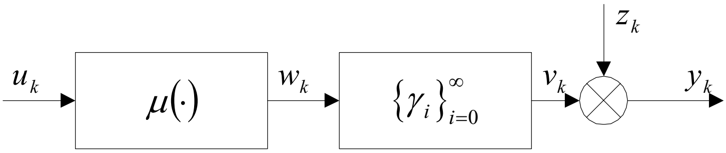


Fig. 1. The identified Hammerstein system

2.2 Assumptions / a Priori Knowledge

1. The input signal $\{u_k\}$ is for $k = ..., -1, 0, 1, ...$ an i.i.d. bounded random process $|u_k| \leqslant u_{\max}$, some $u_{\max} > 0$, and there exists a probability density of u_k, say $\nu(u)$.
2. The nonlinear characteristic $\mu(u)$ is a bounded function on the interval $[-u_{\max}, u_{\max}]$, i.e.
$$|\mu(u)| \leqslant w_{\max} \tag{2}$$
where $w_{\max}$ is some positive constant.
3. The linear dynamics is an asymptotically stable *IIR* filter
$$v_k = \sum_{i=0}^{\infty} \gamma_i w_{k-i} \tag{3}$$
with the unknown impulse response $\{\gamma_i\}_{i=0}^{\infty}$ (such that $\sum_{i=0}^{\infty} |\gamma_i| < \infty$).
4. The output noise $\{z_k\}$ is a random, arbitrarily correlated process, governed by the general equation
$$z_k = \sum_{i=0}^{\infty} \omega_i \varepsilon_{k-i} \tag{4}$$

where $\{\varepsilon_k\}$, $k = ..., -1, 0, 1, ...$, is a bounded stationary zero-mean white noise ($E\varepsilon_k = 0$, $|\varepsilon_k| \leqslant \varepsilon_{\max}$), independent of the input signal $\{u_k\}$, and $\{\omega_i\}_{i=0}^{\infty}$ is unknown; $\sum_{i=0}^{\infty} |\omega_i| < \infty$. Hence the noise $\{z_k\}$ is a stationary zero-mean and bounded process $|z_k| \leqslant z_{\max}$, where $z_{\max} = \varepsilon_{\max} \sum_{i=0}^{\infty} |\omega_i|$.

5. $\mu(u_0)$ is known at some point u_0 and $\gamma_0 = 1$.

As it was explained in detail in [1] and [2], the input-output pair $(u_0, \mu(u_0))$ assumed to be known can refer to arbitrary $u_0 \in [-u_{\max}, u_{\max}]$, and hence we shall further assume for convenience that $u_0 = 0$ and $\mu(0) = 0$, without loss of generality.

2.3 Preliminaries

A fundamental meaning for the methods elaborated in this paper has the dependence between the regression and the nonlinear characteristic

$$R(u) = E\{y_k | u_k = u\} = \gamma_0 \mu(u) + d \text{ where } d = E\mu(u_k) \cdot \sum_{i>0} \gamma_i = \text{const}\,.$$

Since under assumption $\mu(0) = 0$, it holds that $R(0) = d$ and

$$R(u) - R(0) = \gamma_0 \mu(u) \tag{5}$$

The equivalence (5) allows to recover the nonlinear characteristic $\mu()$ under lack of prior knowledge about the linear dynamic subsystem. This observation was successfully utilized in eighties by Greblicki and Pawlak in nonparametric methods, when the parametric form of $\mu()$ is also unknown, but was never explored in parametric identification framework, when some prior knowledge of only $\mu()$ is given.

In this paper we show that the regression-based approach to nonlinearity recovering in Hammerstein system can also be applied when some, even uncertain, parametric prior knowledge of the static characteristic is given.

3 Parametric Approximation of the Regression Function

The methods presented in section 4 recover the true regression function from the input-output measurements. Here we accept the parametric class of model and try to find the best approximation in this class.

3.1 Regression-Based Parametric Approach

In the traditional (purely parametric) approach we suspect that the nonlinear characteristic $\mu(u)$ can be well approximated by the model from the given class

$$\overline{\mu}(u, c), \tag{6}$$

where $c = (c_1, c_2, ..., c_m)^T$ includes finite number of parameters. The function $\overline{\mu}(u, c)$ is assumed to be *differentiable* with respect to c. Let $c^* = (c_1^*, c_2^*, ..., c_m^*)^T$ be the best choice of c in the sense that

$$c^* = \arg\min_c E\left(\mu(u) - \overline{\mu}(u, c)\right)^2. \tag{7}$$

Further, we will explore the generalized version of (6)

$$\overline{\mu}(u, \vartheta) = c_\alpha \overline{\mu}(u, c) + c_\beta,$$

where $\vartheta = \left(c^T, c_\alpha, c_\beta\right)^T$ is the extended model vector enriched with the scale c_α and the offset c_β. Obviously for $c_\alpha = 1$ and $c_\beta = 0$ equation (7) can be rewritten in the form

$$\arg\min_\vartheta E\left(\mu(u) - \overline{\mu}(u, \vartheta)\right)^2 = \left(c^{*T}, 1, 0\right)^T \triangleq \vartheta^*. \tag{8}$$

Remark 1. For functions which are linear in the parameters and has additive constant the classes $\overline{\mu}(u, c)$ and $\overline{\mu}(u, \vartheta)$ are indistinguishable. For example the polynomial model of order m (see [3])

$$\overline{\mu}(u, c) = c_m u^{m-1} + ... + c_2 u + c_1$$

leads to the same class of

$$\overline{\mu}(u, \vartheta) = c_\alpha c_m u^{m-1} + ... + c_\alpha c_2 u + c_\alpha c_1 + c_\beta.$$

Remark 2. If $E\left(\mu(u) - \overline{\mu}(u, \vartheta)\right)^2$ is minimized by $\left(c^{*T}, 1, 0\right)^T$, then $E\left(R(u) - \overline{R}(u, \theta)\right)^2$ is minimized by $\left(c^{*T}, \gamma_0, d\right)^T$.

3.2 Approximation

By rewriting (1) in the form $y_k = \gamma_0 \mu(u_k) + \sum_{i=1}^{\infty} \gamma_i \mu(u_{k-i}) + z_k$, and taking into account that $R(u_k) = \gamma_0 \mu(u_k) + \sum_{i=1}^{\infty} \gamma_i E\mu(u_1)$, we obtain the equivalent (cardinal) description $y_k = R(u_k) + \delta_k$ of the Hammerstein system, in which the total noise

$$\delta_k \triangleq y_k - R(u_k) = \sum_{i=1}^{\infty} \gamma_i \left(\mu(u_{k-i}) - E\mu(u_1)\right) + z_k$$

is zero-mean ($E\delta_k = 0$) and independent of u_k. We want to find the vector θ^* for which the model $\overline{R}(u_k, \theta)$ fits to data the best, in the sense of the following criterion

$$E\left(y_k - \overline{R}(u_k, \theta)\right)^2 = \operatorname{var}\delta_k + E\left(R(u_k) - \overline{R}(u_k, \theta)\right)^2. \tag{9}$$

From (9) we conclude that

$$\arg\min_\theta E\left(y_k - \overline{R}(u_k, \theta)\right)^2 = \arg\min_\theta E\left(R(u_k) - \overline{R}(u_k, \theta)\right)^2,$$

which is fundamental for the least squares approximation

$$\widehat{\theta}_N = \arg\min_\theta \sum_{k=1}^{N} \left(y_k - \overline{R}(u_k, \theta)\right)^2.$$

4 Nonparametric Neural Network Trainer/Teacher

To evaluate $\widehat{R}_N(\overline{u})$ we use a neural network, denoted further by $R_k(u)$, and based on either radial basis [4,5,6], [7, Ch. 17] and [8,9], or wavelet [10,11,12,7, Ch. 18], or classic kernel [13,7, Ch. 5] regression function estimates.

For each pair of the *learning sequence,* (u_k, y_k), $k = 1, 2, \ldots$, and for each point u from some *training set* $\{u_e\}_{e=1}^{N_0}$, the network learning formula is given recursively as:

$$\hat{R}_k\left(u\right) = \hat{R}_{k-1}\left(u\right) + \overbrace{\underbrace{\kappa_k\left(u\right)}_{\text{weight}} \cdot \underbrace{\left[y_k - \hat{R}_k\left(u\right)\right]}_{\text{error}}}^{\text{correction}}, \text{ for all } u = u_e, \tag{10}$$

where (we take $0/0 = 0$ when necessary)

$$\kappa_k\left(u\right) = \frac{\phi_k\left(u\right)}{\hat{f}_k\left(u\right)} \text{ with } \hat{f}_k\left(u\right) = \hat{f}_{k-1}\left(u\right) + \phi_k\left(u\right)$$

and where $\phi_k\left(u\right)$ is a shorthand of the selected kernel function $\phi_{K(k)}\left(u, u_k\right)$, and where, finally, $\hat{f}_k\left(u\right)$ is the recursive estimate of the density of the inputs in the learning sequence. The initial conditions are $\hat{R}_0\left(u\right) = \hat{f}_0\left(u\right) = 0$). The following lemma characterizes the limit properties of the proposed neural network; *cf.* [14].

Lemma 1. *Let the nonlinearity* $R\left(u\right)$ *and the input signal density function have* $\lfloor \nu_R \rfloor$ *and* $\lfloor \nu_f \rfloor$ *derivatives, respectively (for some* $\nu_R, \nu_f > 0$*). If the kernel function* $\phi_{K(k)}\left(u, v\right)$ *has* p *vanishing moments and its bandwidth parameter is governed by the rule* $K\left(k\right) = \left(2\nu + 1\right)^{-1} \log_2 k$, *where* $\nu = \min\left\{\nu_R, \nu_f, p\right\}$, *then, in all* training points $u \in \{u_e\}_{e=1}^{N_0}$ *the network error vanishes and*

$$\left|\hat{R}_k\left(u\right) - R\left(u\right)\right| = \mathcal{O}\left(k^{-\nu/(2\nu+1)}\right), \textit{ in probability.} \tag{11}$$

Proof. See [15] for the proof in case of wavelet network and [14] for the proof for other networks.

5 Model Training and Competition

Let's split the set of *training set* $\{u_e\}$ into two disjoint parts containing $\{u_l\}$ and $\{u_t\}$, being the *learning* and *testing* points, respectively. After each new measurement arrival and application of the recursive update procedure (10)) the training/testing routine is performed on the models $\left\{\overline{R}^{(l)}\left(u, \theta_l\right)\right\}$.

In this phase, the models are trained, *i.e.* their parameters $\{\theta_M\}$ are evaluated using learning part, $\{u_l\}$, of the training points set $\{u_e\}$.

Remark 3. The evaluation routine is particularly simple when the models are linear in parameters and the functions the model is built upon are pairwise orthogonal. Then it actually reduces to solving the linear equation system.

Each model $\overline{R}^{(l)}(u,\theta_l)$ collects its own number of wins W_l. The following two strategies can be now used to model the nonlinearity $\mu(u)$:

1. the *winner-take-all* approach, in which the model with the largest number of wins W_l is selected as the model of the nonlinearity (the "winner-takes-all" approach), *i.e.*
$$R(u,\theta) = \overline{R}^{(l_{\max})}(u,\theta_{l_{\max}}), \text{ such that } l_{\max} = \max_l \{W_l\}$$
or
2. the *soft* approach, in which a *convex combination* of the models is used as the nonlinearity model, *i.e.*
$$R(u,\theta) = \sum_{l=1}^{M} w_l \cdot \overline{R}^{(l)}(u,\theta_l)$$
where $w_l = W_l/k$. Clearly, $\sum_{l=1}^{M} w_l = 1$.

The following theorem describes the limit properties of the proposed model selection algorithm:

Theorem 1. *If the neural network trainer $\hat{R}_k(u)$ converges to the nonlinearity $R(u)$ in all points of the training set, $\{x_e\}$, then the proposed smart model selection algorithm picks (in probability) eventually the best model of the nonlinearity.*

Proof. The proof is immediate. The neural network trainer $\hat{R}_k(u)$ approaches the actual nonlinearity $R(u)$ by virtue of the Lemma 1. Since each model $\overline{R}^{(l)}(u,\theta_l)$ is evaluated to minimize the distance (the error) between models and the estimate, then the convergence rate is determined by the estimate rate, otherwise, the best model converges to the best approximation of the nonlinearity with the same rate.

The theorem says that in both *winner-takes-all* and *soft* strategy, the best nonlinearity model is chosen. Indeed, one can expect that with the growing size of the processed learning sequence pairs (u_k, y_k) the neural network trainer becomes the better-and-better estimate of the unknown nonlinearity, and hence the model closest to the estimate is simultaneously the closest to the nonlinearity, and eventually it 'overwhelms' its rivals. With the number of learning pairs growing to infinity, the weight, w_l, of this model tends to 1 (and the weights of other vanishes).

Remark 4. Clearly, the training set needs to be split into learning and testing parts. If, for instance, we had used use the same set for learning and testing we would have obtained the zero error (the best match) for all models. Consider for example three models utilizing the first $N_0 = 2^\eta$, $\eta = 1, 2, \ldots$, terms of the Haar wavelet, Fourier trigonometric or Legendre polynomial orthogonal series. Assume now that the $\{x_l\}_{l=1}^{N_0}$ are equidistant, *i.e.* they form a binary grid. In such a setting, all these series are orthogonal bases on such a discrete grid and hence are able to represent exactly (recover) any function defined in training points $\{x_l\}$.

6 Simulation Examples

Due to limited number of pages the results of simulation examples are accessible in the full version of the paper, see *http://staff.iiar.pwr.wroc.pl/grzegorz.mzyk*

7 Final Remarks

In the paper we proposed the new algorithms which combines the *parametric* and *nonparametric* approaches to recover the Hammerstein system nonlinearity under *quasi-parametric* prior knowledge. The nonparametric neural network trainer is used first to filter (smooth) the noisy learning sequence, and then to train the model candidates. In the competition phase, the winner is the model which matches best the neural network trainer.

It is shown that in the proposed *winner-take-all* approach, the model which collects the largest number of wins, is selected as the nonlinearity model. Otherwise, in the alternative *soft* strategy, the convex combination of all competitor models is taken as the model. Note, however, that asymptotically both strategies lead to the selection of the single model.

One can point out the following advantages of the algorithm:

1. No need of storing the measurements in memory.
2. Fast model training – the random learning points are replaced by the deterministic ones, *i.e.*, the active experiment techniques (*e.g.* orthogonal plans) can be used in model training in place of the passive ones.
3. Flexible model selection routine – the examine routines, *i.e.* the model competitions can be performed in the user-defined regions of interests, *e.g.* in the working points.
4. The list of model candidates can be open – the new models can join the competition at any time (with a "*wild card*"), and win, if they are actually the proper ones – since all the information about the nonlinearity used to train the models is maintained by the neural network trainer.

Clearly, there are also some weaknesses:

1. The main disadvantage of the proposal consists in its slower convergence rate. It is of nonparametric order, $\mathcal{O}(k^{-1/2+\gamma})$, where $\gamma = 1/2\,(2\nu+1)$, and in fact, is a toll we pay for a smaller prior knowledge. Recall however (*cf.* (11) in the Lemma 1) that, in general, ν grows with the smoothness of the nonlinearity. That is, the smoother the nonlinearity, the smaller γ, and the convergence rate is closer to the typical parametric rate $\mathcal{O}(k^{-1/2})$.
2. The set of model class candidates has to be complete in the sense that the nonlinearity has to belong to the one of them. Otherwise, the nonlinearity can be of any shape and neither model considered in the section 3.2 can be its reasonable approximation.

Remark 5. The algorithm can also be seen as a pattern recognition algorithm classifying the system nonlinearity to the one of the predefined model classes.

The wavelet neural network trainer plays there a role of the raw learning data preprocessor and the tournament routine is an implementation of a nearest-neighbor algorithm; *cf.* [17].

References

1. Hasiewicz, Z., Mzyk, G.: Combined parametric-nonparametric identification of Hammerstein systems. IEEE Transactions on Automatic Control 49(8), 1370–1375 (2004)
2. Hasiewicz, Z., Mzyk, G.: Hammerstein system identification by nonparametric instrumental variables. International Journal of Control 82(3), 440–455 (2009)
3. Śliwiński, P., Rozenblit, J., Marcellin, M.W., Klempous, R.: Wavelet amendment of polynomial models in nonlinear system identification. IEEE Transactions on Automatic Control 54(4), 820–825 (2009)
4. Krzyzak, A., Linder, T., Lugosi, C.: Nonparametric estimation and classification using radial basis function nets and empirical risk minimization. IEEE Transactions on Neural Networks 7(2), 475–487 (1996)
5. Kamiński, W., Strumiłło, P.: Kernel orthonormalization in radial basis function neural networks. IEEE Transactions on Neural Networks 8(5), 1177–1183 (1997)
6. Krzyzak, A., Linder, T.: Radial basis function networks and complexity regularization in function learning. IEEE Transactions on Neural Networks 9(2), 247–256 (1998)
7. Györfi, L., Kohler, M., Krzyżak, A., Walk, H.: A Distribution-Free Theory of Nonparametric Regression. Springer, New York (2002)
8. Buhmann, M.D.: Radial Basis Functions: Theory and Implementations. Cambridge University Press, Cambridge (2003)
9. Ferrari, S., Maggioni, M., Borghese, N.A.: Multiscale approximation with hierarchical radial basis functions networks. IEEE Transaction one Neural Networks 15(1) (2004)
10. Zhang, Q., Benveniste, A.: Wavelet networks. IEEE Transactions on Neural Networks 3, 889–898 (1992)
11. Zhang, J., Walter, G., Miao, Y., Lee, W.N.: Wavelet neural networks for function learning. IEEE Transactions on Signal Processing 43, 1485–1497 (1995)
12. Hasiewicz, Z.: Modular neural networks for non-linearity recovering by the Haar approximation. Neural Networks 13, 1107–1133 (2000)
13. Greblicki, W., Pawlak, M.: Identification of discrete Hammerstein system using kernel regression estimates. IEEE Transactions on Automatic Control 31, 74–77 (1986)
14. Rutkowski, L.: Generalized regression neural networks in time-varying environment. IEEE Transactions on Neural Networks 15(3), 576–596 (2004)
15. Śliwiński, P., Hasiewicz, Z.: Recursive wavelet estimation of Hammerstein systems nonlinearity. International Journal of Control (2010) (in preaparation)
16. Greblicki, W., Mzyk, G.: Semiparametric approach to Hammerstein system identification. In: Proceedings of the 15th IFAC Symposium on System Identification, Saint-Malo, France, pp. 1680–1685 (2009)
17. Rutkowski, L.: Adaptive probabilistic neural networks for pattern classification in time-varying environment. IEEE Transactions on Neural Networks 15(4), 811–827 (2004)

Recent Progress in Applications of Complex-Valued Neural Networks

Akira Hirose

Department of Electrical Engineering and Information Systems, The University of Tokyo, Japan
ahirose@ee.t.u-tokyo.ac.jp
http://www.eis.t.u-tokyo.ac.jp/

Abstract. In this keynote speech, we present recent progress in the complex-valued neural networks by focusing on their applications.

The most significant advantage of complex-valued neural networks originates not from the two-dimensionality of the complex plane, but from the fact that the multiplication of the synaptic weights, which is the elemental process at the synapses of neurons that construct the whole network, produces the modulation in two types of signal entities, that is, phase rotation and amplitude increase / decrease [9] [11] [10]. In other words, the function of the whole network consists of the phase rotation as well as the amplitude modulation. This fact reduces the degree of freedom in learning and self-organization, in comparison with a double-dimensional real-valued neural network, so that the network yields learning or self-organization results that agree with phase-rotational phenomenon or information.

Accordingly, the complex-valued neural networks are most suitable for robotics having rotational articulation, wave information processing where the phase carries primary information, etc. Complex-valued neural networks (CVNNs), therefore, extend the application fields steadily [8] . We have various application systems employing CVNNs in the field of, for example, ultrasonic fault detection to find defects in metals and other materials [2] , blind separation based on principal component analysis (PCA) in sonar [31] and voice processing [23] , radars including ground penetrating radars to visualize plastic landmines [6] [7] [30] [19] [20] [21] and satellite radars to estimate landscape information [29] and / or land-use classification [24] , blur-compensation image processing [1] , filtering and other time-sequential signal processing [4] [5], frequency-domain multiplexed microwave signal processing [13] and pulse beamforming in ultra-wideband (UWB) communications [25], frequency-domain multiplexed neural networks and learning logic circuits using lightwave [8] [14] [16] and fast adaptive three-dimensional holographic movie generation for optical tweezers [15] [27] , and developmental learning of motion control in combination with reinforcement learning [12] . In parallel, general associative memories [17] and independent component analysis (ICA) neural networks [22] [18] are also making progress in their improvement.

We often use the complex-valued least mean square (LMS) algorithm in linear processing with a simple network structure [28] . Neural networks in general conduct nonlinear processing. Regarding the nonlinearity to be employed, we have a series of discussions including several milestone papers [3] . The pros and cons of respective nonlinearities basically depend on the nature of the signal to be treated. We often

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 42–46, 2010.
© Springer-Verlag Berlin Heidelberg 2010

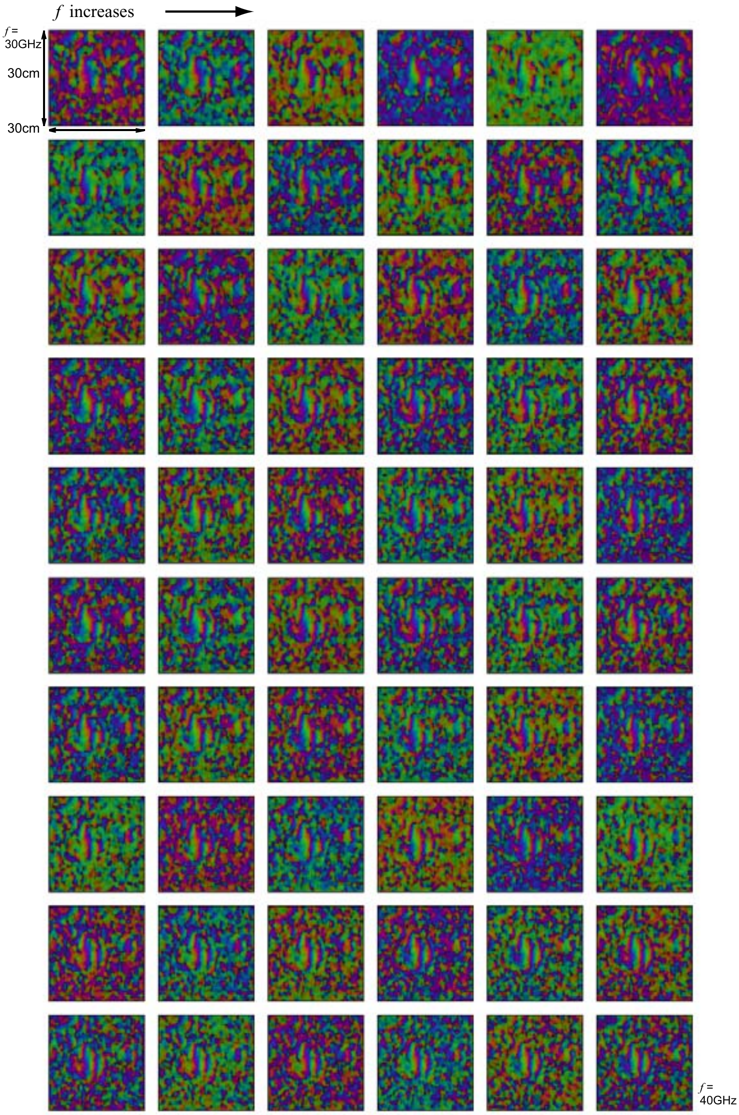


Fig. 1. Multi-frequency observation data for the plastic landmine buried in the ground at the center of 30×30 cm^2 area. In this early-stage experiment to find a plastic landmine buried very shallowly, the radar frequency f was stepped from 30GHz (top left) to 40GHz (bottom right) with a constant interval of about 0.16GHz. Brightness shows intensity, while hue presents phase [6].

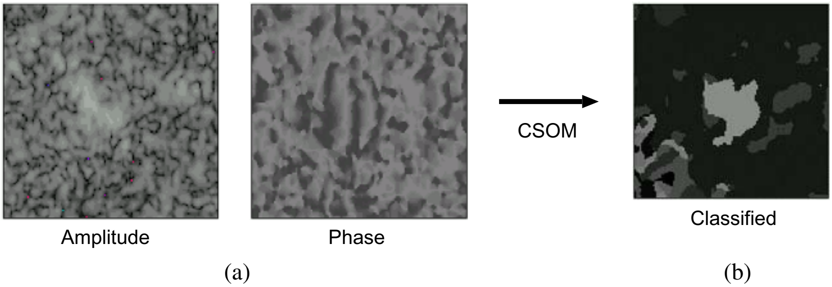


Fig. 2. (a)Sample amplitude and phase data at a frequency out of ten frequency-point data fed to the CSOM, shown in gray scale separately, and (b)CSOM classification result of the plastic mine buried near the ground surface at the center of the area [6].

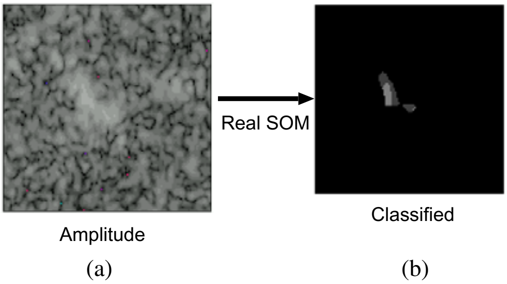


Fig. 3. (a)Sample amplitude data at a frequency out of ten frequency-point data fed to the real-SOM, shown in gray scale, and (b)real-SOM classification result of the plastic mine buried near the ground surface at the center of the area, for comparison with the CSOM case [6].

deal with wave-related complex signals [9] . When we observe a wave signal by using coherent detection, or a baseband complex signal generated through Hilbert transform, we obtain the complex-amplitude, i.e., the phasor, inevitably. The CVNNs are compatible with such wave phenomena. This is the most significant feature of the CVNNs. Actually, in the very early stage of the CVNN research, a pioneering idea and a basic experiment was reported concerning this important feature. That is, in 1992, M.Takeda & T.Kishigami pointed out the fact that the electromagnetic field in a phase-conjugate resonator is formulated in the same manner as that of an associative memory, and that the resonant system realizes a quite fast recall [26] . In this case, the limitation in the energy supply causes amplitude saturation, which realizes the neural nonliearity in the signal amplitude in a natural way.

In such wave-information processing or wave control, it is essentially important to deal directly with phase (or phase difference) and amplitude. The reason lies in the facts that the amplitude corresponds to the wave energy (e.g., number of photons of lightwave), and that the phase difference represents time course and/or position change. From this viewpoint, the so-called amplitude-phase-type nonlinearity is consistent with the wave [3] [8] [9] , as is often the case in signal processing widely in electronics.

Signal processing in radar imaging is an important example of applications. Figures 1, 2 and 3 illustrate the image processing in a plastic landmine visualization system we developed previously [6]. Every square in Fig.1 shows 30cm×30cm two-dimensional image of electromagnetic wave scattered and/or reflected by soil, that contains a plastic landmine as well as stones and clods, measured at multiple frequencies. Brightness shows the intensity, while hue represents the phase. In this raw data, we cannot tell the location where the landmine is buried. In the system, we first extract local textural features, i.e., textures in space and frequency domains, by calculating correlation coefficients in the space and frequency domains locally. Then we feed a set of obtained feature vectors to a complex-valued self-organizing map (CSOM) so that the CSOM classifies the feature vectors adaptively. By mapping back the obtained classes onto two-dimensional square in the space domain, we acquire a segmented map as shown in Fig.2, where we find a landmine at the center. When we employ a real-valued self-organizing map, the result is very different as shown in Fig.3, in which we find that the visualization failed.

In this keynote speech, we review the recent progress in the complex-valued neural networks with emphasis on applications. We find that many applications exhibit respective strength specific to complex-valued neural networks where the phase information plays an important role in the learning and self-organizing dynamics.

References

1. Aizenberg, I., Paliy, D.V., Zurada, J.M., Astola, J.T.: Blur identification by multilayer neural network based on multivalued neurons. IEEE Transactions on Neural Networks 19(5), 883–898 (2008)
2. Birx, D.L., Pipenberg, S.J.: A complex mapping network for phase sensitive classification. IEEE Transactions on Neural Networks 4(1), 127–135 (1993)
3. Georgiou, G.M., Koutsougeras, C.: Complex domain backpropagation. IEEE Transactions on Circuits and Systems II 39(5), 330–334 (1992)
4. Goh, S., Mandic, D.P.: Nonlinear adaptive prediction of complex valued nonstationary signals. IEEE Transactions on Signal Processing 53(5), 1827–1836 (2005)
5. Goh, S., Mandic, D.P.: An augmented extended kalman filter algorithm for complex-valued recurrent neural networks. Neural Computation 19(4), 1–17 (2007)
6. Hara, T., Hirose, A.: Plastic mine detecting radar system using complex-valued self-organizing map that deals with multiple-frequency interferometric images. Neural Networks 17(8-9), 1201–1210 (2004)
7. Hara, T., Hirose, A.: Adaptive plastic-landmine visualizing radar system: effects of aperture synthesis and feature-vector dimension reduction. IEICE Transactions on Electronics E88-C(12), 2282–2288 (2005)
8. Hirose, A.: Applications of complex-valued neural networks to coherent optical computing using phase-sensitive detection scheme. Information Sciences –Applications–2, 103–117 (1994)
9. Hirose, A.: Complex-Valued Neural Networks. Springer, Heidelberg (2006)
10. Hirose, A.: Complex-valued neural network. Video archive (June 14, 2009), `http://sites.google.com/site/ciseducationsite/home/video-tutorials-produced-by-cis/`
11. Hirose, A.: Complex-valued neural networks: The merits and their origins. In: Proceedings of the Internatinal Joint Conference on Neural Networks (IJCNN), Atlanta, June 14–19, pp. 1237–1244. IEEE / INNS (2009)

12. Hirose, A., Asano, Y., Hamano, T.: Developmental learning with behavioral mode tuning by carrier-frequency modulation in coherent neural networks. IEEE Transactions on Neural Networks 17(6), 1532–1543 (2006)
13. Hirose, A., Eckmiller, R.: Behavior control of coherent-type neural networks by carrier-frequency modulation. IEEE Transactions on Neural Networks 7(4), 1032–1034 (1996)
14. Hirose, A., Eckmiller, R.: Coherent optical neural networks that have optical-frequency-controlled behavior and generalization ability in the frequency domain. Applied Optics 35(5), 836–843 (1996)
15. Hirose, A., Higo, T., Tanizawa, K.: Efficient generation of holographic movies with frame interpolation using a coherent neural network. IEICE Electronics Express 3(19), 417–423 (2006)
16. Kawata, S., Hirose, A.: Frequency-multiplexing ability of complex-valued Hebbian learning in logic gates. International Journal of Neural Systems 12(1), 43–51 (2008)
17. Lee, D.L.: Improvements of complex-valued Hopfield associative memory by using generalized projection rules. IEEE Transactions on Neural Networks 17(5), 1341–1347 (2006)
18. Li, H., Adali, T.: A class of complex ICA algorithms based on the kurtosis cost function. IEEE Transactions on Neural Networks 19(3), 408–420 (2008)
19. Masuyama, S., Hirose, A.: Walled LTSA array for rapid, high spatial resolution, and phase sensitive imaging to visualize plastic landmines. IEEE Transactions on Geoscience and Remote Sensing 45(8), 2536–2543 (2007)
20. Masuyama, S., Yasuda, K., Hirose, A.: Multiple mode selection of walled-ltsa array elements for high resolution imaging to visualize antipersonnel plastic landmines. IEEE Geoscience and Remote Sensing Letters 5(4), 745–749 (2008)
21. Nakano, Y., Hirose, A.: Improvement of plastic landmine visualization performance by use of ring-csom and frequency-domain local correlation. IEICE Transactions on Electronics E92-C(1), 102–108 (2009)
22. Novey, M., Adali, T.: Complex ICA by negentropy maximization. IEEE Transactions on Neural Networks 19(4), 596–609 (2008)
23. Sawada, H., Mukai, R., Araki, S., Makino, S.: Polar coordinate based nonlinear function for frequency-domain blind source separation. IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Sciences E86A, 590–596 (2003)
24. Suksmono, A.B., Hirose, A.: Adaptive complex-amplitude texture classifier that deals with both height and reflectance for interferometric sar images. IEICE Transaction on Electronics E83-C(12), 1905–1911 (2000)
25. Suksmono, A.B., Hirose, A.: Beamforming of ultra-wideband pulses by a complex-valued spatio-temporal multilayer neural network. International Journal of Neural Systems 15(1), 1–7 (2005)
26. Takeda, M., Kishigami, T.: Complex neural fields with a hopfield-like energy function and an analogy to optical fields generated in phase-conjugate resonators. Journal of Optical Society of America A 9(12), 2182–2191 (1992)
27. Tay, C.S., Tanizawa, K., Hirose, A.: Error reduction in holographic movies using a hybrid learning method in coherent neural networks. Applied Optics 47(28), 5221–5228 (2008)
28. Widrow, B., McCool, J., Ball, M.: The complex lms algorithm. Proceedings of the IEEE 63, 719–720 (1975)
29. Yamaki, R., Hirose, A.: Singular unit restoration in interferograms based on complex-valued Markov random field model for phase unwrapping. IEEE Geoscience and Remote Sensing Letters 6(1), 18–22 (2009)
30. Yang, C.C., Bose, N.: Landmine detection and classification with complex-valued hybrid neural network using scattering parameters dataset. IEEE Transactions on Neural Networks 16(3), 743–753 (2005)
31. Zhang, Y., Ma, Y.: CGHA for principal component extraction in the complex domain. IEEE Transactions on Neural Networks 8(5), 1031–1036 (1997)

Hybrid-Maximum Neural Network for Depth Analysis from Stereo-Image

Łukasz Laskowski

Technical University of Czestochowa, Department of Computer Engineering,
Al. A.K. 36, 42-200 Czestochowa, Poland

Abstract. In present paper, we describe completely innovation architecture of artificial neural nets based on Hopfield structure for solving of stereo matching problem. Hybrid neural network consists of classical analogue Hopfield neural network and maximal neural network. The role of analogue Hopfield network is to find of attraction area of global minimum, whereas maximum network is to find accurate location of this minimum. Presented network characterizes by extremely high rate of working with the same accuracy as classical Hopfield-like network. It is very important as far as application and system of visually impaired people supporting are concerned. Considered network was taken under experimental tests with using real stereo pictures as well simulated stereo images. This allows on calculation of errors and direct comparison to classic analogue Hopfield neural network. Results of tests have shown, that the same accuracy of solution as for continuous Hopfield-like network, can be reached by described here structure in half number of classical Hopfield net iteration.

Keywords: Hopfield, neural networks, stereovision, depth analysis, hybrid network.

1 Introduction

Sight is the sense that people make most use of it in everyday life. Eyes give us possibility to estimating distance to nearby object, recognizing, reading, predicting of moving object position and so on. All of these guarantee safe and convenient life. People, who lost or damaged this sense are not able to live independently. It is a multi-disciplinary effort to develop devices for individuals whom happened to loose their sight. One of solution can be portable binocular vision system, which makes possible in orientation and mobility for blind users. The system can be based on stereovision [1]. The advantages of stereovision include ease of use, non-contact, non-emission, low cost and flexibility. For these reasons, it focuses attention of scientists to develop of stereo-vision methods.

Stereovision is natural way of determination of distance by human. The simplified model of human sight can be present as two parallel cameras, and this model (named parallel stereovision system) is going to be considered in present paper. In stereovision system (two displaced parallel cameras) expression on distance to given world point can be written as eqn.1.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 47–55, 2010.
© Springer-Verlag Berlin Heidelberg 2010

$$z = \frac{df}{x'_l - x'_r}. \tag{1}$$

Where x'_l and x'_r are positions of point's image on planes of left and right cameras, f denotes the focal length of the cameras, and d represents the distance between two cameras.

Depth analysis [2] depends on determination of third dimension of each point in observed scene. Third dimension of point is inversely proportional to difference in position of image of this point on plane left and right (disparity). Problem seems to be trivial and is trivial for one point. Real scene contains from large amounts of points, and the complexity of the correspondence problem depends on the complexity of the scene. There are constraints and schemes that can help reduce the number of false matches, but many unsolved problems still exist in stereo matching [3]. Main problems are occlusion, discontinuity of depth, discontinuity of periphery as well as regularity and repetitivity. For this reasons stereo matching problem is one of the most complex problems in computer vision.

There were purposed few types of algorithms for solving this problem [4]. Main of this was: Feature based algorithms, Phase based algorithms, energy based algorithms and area based algorithms. Nowadays algorithms in original form are used rarely only to very basic problems. Scientists try to merge different types of solution in order to pull out as many advantages from all types of algorithms as possible and avoid disadvantages. Energy can be minimized also by using Hopfield-like Neural Nets [5], [6]. The ability of the Hopfield network to solve optimization problems relies on its steepest descent dynamics and guaranteed convergence to local minima of the energy landscape. This kind of system was used in stereo-matching problem. Both types of Hopfield-like network were used: continuous and discrete.

Looking on the state-of-the-art-of stereovision matching with using of Hopfield-like networks one can has the impression that this domain is well investigated. However this subject is so wide and complicated, that there are still possibility of improve efficiency such systems or working out better architecture of nets. Author tried to use mentioned above solutions to stereo matching process. Each time error of network working (calculation of errors was described in farther chapters of present work) and number of iteration were noted. Unfortunately neither of presented above network works correctly. Error of working was very high (each time above 30%) what practically eliminate those methods to solving of stereo matching problem. It is the reason of looking for new solution and new architecture of neural networks. Author tried to increase of working ratio with no loosing of solution accuracy.

In present paper completely innovative architecture of network to solving stereo matching problem is presented. Due to using of two types of networks: classical analogue Hopfield like network and maximum network described here structure is working much faster then classical Hopfield net with satisfied accuracy. Its efficiency was confirmed in tests on real and simulated pictures (what allowed on error calculation).

2 Analogue Hopfield Neural Network for Stereo Matching Problem

Stereo matching problem can be casted as an optimization task, where an energy function, representing constrains on the solution, has to be minimized. The optimization problem then can be performed by means of the Hopfield neural network. The most accurate solution can be obtained by analogue Hopfield-like neural net. The proposed network consists of $n \times n$ neurons for one epipolar line in image. It is easy to note that target system will consist of n networks working pararely - each network will realize stereo-matching problem for one epipolar line. n is dimension of images ($width = height = n$). Each neuron neu_{ik} with potential v_{ik} is responsible for fitting i-point in right image to k-point in left image. The highest external potential of neu_{ik} is, the better fittingo of points. In final configuration only for corresponding points, i in right image to k in left image potential of neu_{ik} will be equal 1, for rest point external potential of neurons will be equal 0. Very convenient is to represent neurons an a matrix, named Fitting Matrix (FM), where number of row represent i index, and number of column represents k index.

The equation of motion of the Hopfield model must be discretized by means of a numerical method. In this case Euler discretization was used:

$$u_{ik}(t+1) = u_{ik}(t) + \Delta t\Big(\sum_{j}\sum_{l} t_{ik,jl} v_{jl}(t) + I_{ik} - \frac{u_{ik}(t)}{\tau}\Big) \tag{2}$$

where Δt is time step, τ is a positive constant (interpreted as neuron relaxation time). In presented design value $\Delta = 10^{-3}$ have been chosen, which have been determined to be small enough for the Euler rule to provide enough accuracy.

3 The Energy Function

In presented here method crucial is the energy function [7]. Network proceeds minimization of this function until it finds minimum. This means that solution of problem was found.

Minimization of energy function must secure following criterions:

1. For couples of correlated points (i, k) and (j, l) in given epipolar line, where i and j are numbers of point in right image, k and l are numbers of point in left image, Correlation Coefficient $C_{ik,jl}$ should have as high value, as possible - term of *Correlation*;
2. Assigning must be reciprocally unique - term of *Uniqueness*;
3. Sequence of assigning in areas must be kept - term of *Area Sequence*;
4. Continuity of assigning in areas must be kept - term of *Continuity*;
5. Global sequence of assigning must be kept - term of *Global Sequence*;

Taking all these terms into consideration, energy function can be expressed in form of equation:

$$
\begin{aligned}
\mathrm{E}= & -a\sum_i\sum_k\sum_j\sum_l C_{ik,jl}v_{ik}v_{vjl} \\
& +b\big(\sum_i\sum_k\sum_{l\neq k} v_{ik}v_{jl}+\sum_i\sum_k\sum_{j\neq i} v_{ik}v_{jl}\big) \\
& +c\sum_i\sum_k\sum_{l\leq k} v_{ik}v_{(i+1)l}\sigma_{i,i+1}+d\sum_i\sum_k\sum_l v_{ik}v_{(i+1)l}\sigma_{i,i+1}\xi_{ik,jl}
\end{aligned}
\tag{3}
$$

where a, b, c, d, e are weight coefficients of each energy components. Having an energy function given as eqn. 3 interconnection weights and external currents are given following equation:

$$
\begin{cases}
t_{ik,jl}=aC_{ik,jl}-b\delta_{ij}(1-\delta_{kl})-b\delta_{kl}(1-\delta_{ij})-c\rho_{l<k}\delta_{(i+1)j}\sigma_{i,i+1} \\
\qquad\quad -d\delta_{(i+1)j}\sigma_{i,i+1}\xi_{ik,jl} \\
I_{ik}=0
\end{cases}
\tag{4}
$$

where δ_{ij} is Kronecker delta, sign $\rho_{l<k}$ is defined by following equation:

$$
\rho_{i<k}=\begin{cases} 0 \text{ for } l>k; \\ 1 \text{ for } l\leq k; \end{cases}
\tag{5}
$$

Values, given by eqn. 4, are basis of working of continuous Hopfield neural network.

4 Architecture of Hybrid-Maximum Neural Network

Disadvantage of analogue Hopfield neural network is long time of computation. Rapidity of working is very important as far as target system is concerned - it should work in real-time. Much faster is a maximum neural network. Additional advantage of maximum network is automatically meeting of term of uniqueness. This type of neural structure was introduced by Takefuji and al in [8]. Maximum neural network was defined as discrete Hopfield-like network with specific activation function: only neuron with the highest value of internal potential (in some group) is activated, rest of neurons have low potential. The maximum activation function for stereo matching problem can be formulated as follow:

$$
f(u_{ij})=\begin{cases} 1 \text{ if } u_{ij}=max(u_{i1},u_{i2},...,u_{in}) \\ 0 \text{ otherwise}; \end{cases} \qquad i,j=1,...,n.
\tag{6}
$$

This kind of neural network found application in optimization problems [9]. Unfortunately in original form maximum network is not fit to solving of stereo-matching problem. The reason is the same as for discrete Hopfield neural network - stereo matching problem is too complex and network is trapped in local minima.

However it is possible to join precision of working of analogue Hopfield network with rapidity of maximum neural network's working. Presented here hybrid neural network contains of analogue Hopfield network and maximum neural network. An architecture of discussed here neural network was depicted on fig. 1.

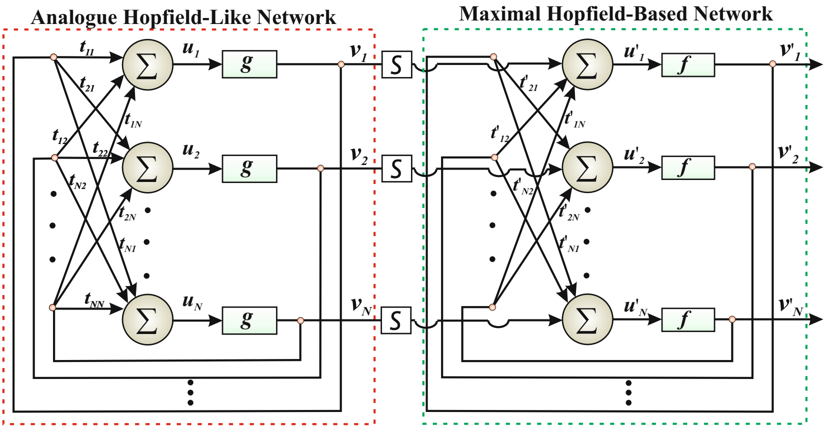


Fig. 1. An architecture of hybrid - maximum neural network

In first stage of working analogue Hopfield neural network is looking for attraction area of global minimum. After finding attraction area of global minimum, network is switched to maximum mode thanks to block of switched function S (see fig. 1). Switching follows after given number of iteration (determined empirically). In maximum mode network evaluates fast towards towards global minimum and term of uniqueness is kept automatically thanks to maximum activation function. It must be stressed, that the form of energy function for maximum network is little different, as on eqn. 4 - there is only one term of uniqueness, second is realized by activation function.

5 Experimental Results

The proposed method was implemented on a Personal Computer with Pentium IV-2.80 GHz CPU and 2 GB SDRAM.

Helpful to analysis of network working is the neurons activity map. It can be interpreted as a graphical form of fitting matrix for investigated line - white points mean neurons with high potentials, black points correspond to neurons with low potentials. Intermediate colors correspond to values between 0 and 1. Thanks to neurons activity map the dynamics of neural network can be observed (map is updated with each iteration).

The resolution of input stereo-images is 100 ×100. Images are calibrated in order to find corresponding lines, before starting stereo matching procedure. This process allows on scanning of pictures line-by-line, what decreases complexity of method.

To verify the efficiency of the proposed method, an experiment was performed using both simulated and real images. Using simulated images allowed on error calculation (only in this case error is calculated).

An operational procedure for solving the stereo matching problem is summarized as follows:

1. Assume number of image epipolar line $h = 0$;
2. Assume maximum number of iteration it_{max} enough to finding attraction area of global minimum;
3. IEF mapping into Analogue Hopfield Network:
   (a) Compute external inputs of neurons and their interconnection strengths using eqn. 4 (with keeping of symmetrical interconnection strength's matrix);
   (b) Initialize states of neurons in heuristic way - assume potentials v_{ik} proportional to correlation coefficients C_{ik};
4. Continuous Hopfield Network updating procedure for energy minimization (working in continuous Hopfield mode):
   (a) For each neuron compute the internal potential, with using of eqn. 2;
   (b) For each neuron compute the external potential using sigmoidal activation function;
   (c) If number of iteration is equal it_{max}, go to (5), else go to (a);
5. Energy function mapping into Maximum Network:
   (a) Compute external inputs of neurons and their interconnection strengths (with keeping of symmetrical interconnection strength's matrix);
   (b) Assume states of neurons the same as at the end of working of continuous Hopfield network;
6. Maximum Network updating procedure for energy minimization (working in maximum mode):
   (a) For each neuron compute the internal potential;
   (b) For each neuron compute the external potential using eqn. 6;
   (c) If changes of internal potentials for each neurons are equal zero go to (7), else go to (a);
7. If present epipolar line is not last one, increment number of line $h = h + 1$ and go to (2), else go to (8);
8. End simulation.

There was assumed, that attraction area of global minimum usually reached after 50 iterations (empirically confirmed). In maximum mode stable state is reached after at most 20 iterations, and this limit of iteration was assumed in order to have possibility of confirmation of results reached for different stereo-images.

Results of stereo matching process carried out by Hybrid Maximum Network can be seen below, for simulated images (fig. 2), and for real images (fig. 3).

In first row stereo pictures, used for stereo matching process and obtained depth map were shown. Second row shows neurons activity maps for 80 scanning line (arbitrary assumed) in iterations (number of "n") of neural net working in analogue Hopfield mode. The last row presents the same sequence, repeated for network working in maximum mode.

The error of stereo matching process can be calculated only for simulated pictures (possibility of neurons activity map determination). Results of Hybrid Maximum Network's working was juxtaposed with results of stereo matching

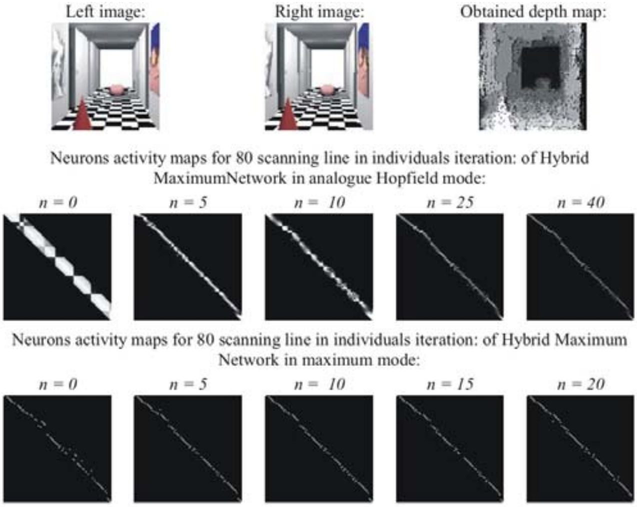

Left image:
Right image:
Obtained depth map:
Neurons activity maps for 80 scanning line in individuals iteration: of Hybrid MaximumNetwork in analogue Hopfield mode:
n = 0
n = 5
n = 10
n = 25
n = 40
Neurons activity maps for 80 scanning line in individuals iteration: of Hybrid Maximum Network in maximum mode:
n = 0
n = 5
n = 10
n = 15
n = 20

Fig. 2. The result of stereo matching process caried out by Hybrid-Maximum Neural Network for simulated stereo-images

process done by analogue Hopfield-like network and Maximum network. Considered here network types worked in the same conditions, with using the same fitting coefficients. Errors of working (δd) and iteration numbers, necessary to reaching stable state amounted:

- $\delta d = 20.04\%$, 60 iterations for Hybrid Maximum network,
- $\delta d = 19.89\%$, 83 iterations for analogue Hopfield-like network,
- $\delta d = 68.52\%$, 54 iterations for Maximum Network.

Analyzing of network working (fig. 2 and fig. 3) one can notice, that in analogue Hopfield mode fitting is ununiqueness. This can be concluded by analyzing of neurons activity maps. In the case of uniqueness of stereo matching in each row and each column of fitting matrix (its graphical form is neurons activity map) should be at very most one non-zero element, whereas in stable state few non-vanishing elements can be observed in columns and rows of fitting matrix. Because of ununiqueness it is different to stay anything about sequence in areas. Also term of keeping of depth continuity in areas can not be stayed. This mistake can be corrected in maximum working mode. Maximum activation function involve meeting of uniqueness term, what can be seen in iterations of network in maximum mode. In each line of FM only on non-zero element can be seen. Stable state is reached very fast thanks to limitation of possible neuron states configuration (maximum activation function). In comparison to analogue Hopfield network, Hybrid Maximum Network is working with the same efficiency (error about 20% in both cases), but much faster. Results of stereo matching obtained by maximum network can not be accepted for the sake of large error - above 68% what discredit this kind of network to solving stereo matching task.

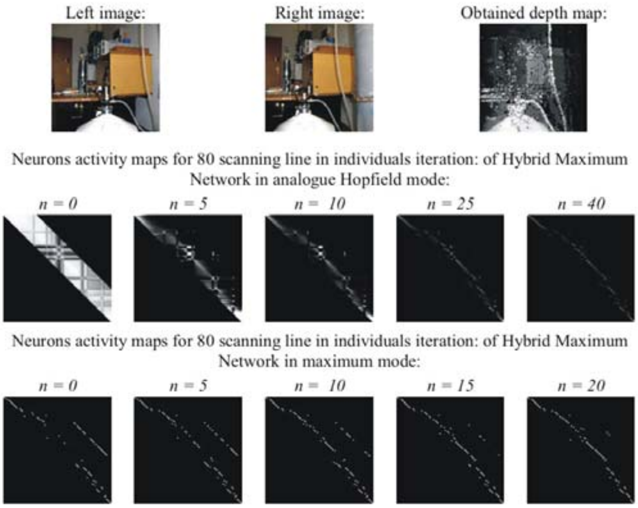

Left image:
Right image:
Obtained depth map:
Neurons activity maps for 80 scanning line in individuals iteration: of Hybrid Maximum Network in analogue Hopfield mode:
n = 0
n = 5
n = 10
n = 25
n = 40
Neurons activity maps for 80 scanning line in individuals iteration: of Hybrid Maximum Network in maximum mode:
n = 0
n = 5
n = 10
n = 15
n = 20

Fig. 3. The result of stereo matching process caried out by Hybrid-Maximum Neural Network for real stereo-images

6 Conclusion

This study presents using of innovative architecture of Hopfield based neural network-Hybrid Maximum Network. Introduced here network has been used to stereo matching process. The stereo correspondence problem has been formulated as an optimization task where an energy function of network is minimized. The advantage of using a Hopfield neural network is that a global match is automatically achieved because all the neurons are interconnected in a feedback loop so that the output of one affects the input of all the others. The convergence to stable state is guaranteed for continuous Hopfield-like network with continuous activation function. The parallel execution capability of this structure is also a powerful reason taking under consideration target system assisting aged people and/or visually impaired. Additionally thanks to using of maximum mode, time of computation significantly decreases.

The experimental results indicate significant gains from using of maximum mode after finding of global minimum's attraction area. A comparative analysis, performed with the classical Hopfield network, maximum network and Hybrid Maximal Network indicated beter performance of last type of network. Obtained solution of stereo correspondence problem was similar to obtained by analogue Hopfield-like network, but reached in less amount of iterations.

References

1. Aleksander, I.: Artificial vision for robots. Korgan Page (1983)
2. Barlow, H.B., Blackmore, C., Pettigrew, J.D.: The natural mechanism of binocular depth discrimination. J. Physiology 193, 327–342 (1967)

3. Faugeras, O.: Three-dimensional computer vision. A geometric viewpoint. MIT Press, Cambridge (1993)
4. Alvarez, L.: Dense Disparity Map Estimation Respecting Image Discontinuities: A PDE and Scale-Space Based Approach. Journal of Visual Communication and Image Representation 13, 3–21 (2002)
5. Hopfield, J.J., Tank, D.W.: Neural computation of decisions in optimization problems. Biological Cybernetics 52, 141–152 (1985)
6. Hopfield, J.J., Tank, D.W.: Artificial neural networks. IEEE Circuits and Devices Magazine 8, 3–10 (1988)
7. Amit, D.J., Gutfreung, H., Sampolinsky, H.: Spin-glass models of neural networks. Physical Review A2 32(2), 1007–1018 (1985)
8. Takefuji, Y., Lee, K.C., Aiso, H.: An artificial maximum neural network: a winner-take-all neuron model forcing the state of the system in a solution domain. Biological Cybernetics 67(3), 243–251 (1992)
9. Takefuji, Y., Lee, K.C.: Neural network computing for knight's tour problems. Neurocomputing 4, 249–254 (1992)

Towards Application of Soft Computing in Structural Health Monitoring

Piotr Nazarko and Leonard Ziemiański

Rzeszow University of Technology, Department of Structural Mechanics,
W. Pola 2, 35-959 Rzeszów, Poland
{pnazarko,ziele}@prz.edu.pl
http://www.prz.edu.pl/

Abstract. The paper presents preliminary results of data analysis and discusses the application of soft computing methods in the field of non-destructive tests. The main objective of developed diagnostic system are the automatic detection and evaluation of damage. Thus the system is composed of two signal processing techniques known as novelty detection and pattern recognition. For this purpose autoassociative as well as feed-forward neural networks are used. All the signals used for training the system are obtained from laboratory tests of strip specimens, where phenomenon of elastic wave propagation in solids was utilized. Computed parameters of time signals defines various types of input vectors used for training neural networks. The results finally obtained prove that the proposed diagnostic system made automation of structure testing possible and can be applied to Structural Health Monitoring.

Keywords: Neural networks, novelty detection, damage evaluation, elastic waves, signal processing, structural health monitoring.

1 Introduction

The assessment of structural integrity and early failure detection are essential concerns for engineers from many branches of industry. Improved structure safety, reduction of maintenance costs and increased structure efficiency are possible nowadays thanks to the continuous development of Structural Health Monitoring (SHM) systems [1,2]. Such systems transform information related to quantitative variation in the parameters measured and should indicate the current state of the monitored structure. The amount of data is usually quite large and its analysis requires application of efficient algorithms consisting of signal processing, feature extraction and inference rules. At present it is very important, especially with regards to aircraft structures, to satisfy the demand for diagnostic systems operating *on-line*.

An equally important issue during the inspection of large structures is the human factor, since lengthy and tiresome tests may result in undetected damage. This is why SHM system integrated with the monitored structure is invaluable, allowing inspection frequency to be increased.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 56–63, 2010.
© Springer-Verlag Berlin Heidelberg 2010

One method that effects automatic analysis of the signals measured is related to *soft computing* [3] and neural networks (NNs) are most commonly used for this purpose. Their success is due to their ability to learn an unknown relation between the input and output vectors. Each input and output vector pair defines a pattern, while the complete set of patterns defines the database used for learning, testing and validation of the NN. Another important feature of NNs is an ability to generalize which manifests itself in the production of a meaningful output for data previously unseen in the training process. In this manner the trained NN can then predict damage parameters for new samples that were not included in the learning sample set.

After successful training the NN can be used as a hardware unit for the purpose of structure diagnosis. Structures or their elements may be equipped with such units and analysis of the measured signals can be performed automatically, eliminating tedious and time-consuming inspections. Miniaturization of electronic devices, solutions for energy harvesting and modern NDT techniques have all contributed to the recent development of SHM systems.

2 Intelligent Damage Detection

Damage identification is recognized as a hierarchical structure of certain precision levels consisting of detection, localization, assessment, etc. [1]. The first level indicates qualitatively that damage may be present in the structure and it can be detected without prior knowledge of how the system will behave when damaged. Further levels provide information about probable position of the damage and estimate its type or extent.

One approach to damage identification is based on the idea of pattern recognition. Such an algorithm assigns a class label to a sample of measured data. The appropriate class labels encode damage type, location, etc. Each possible fault class will usually have a training set of measurement vectors that are associated uniquely with it. An algorithm that works by training are NNs. The type of learning procedure in which the diagnostic is trained by showing it the desired label for each data set is called *supervised learning*. Typically it requires many presentations of data, and the definition of such data involves a huge effort in both computational and experimental investigations. Obviously it is very convenient when the patterns used for the learning diagnosis are computed numerically and then the system is validated using patterns obtained from real structures or laboratory specimens. There are some situations in which the numerical model of the structure does not exist or lacks the necessary precision to investigate certain phenomena (e.g. composite materials, fatigue, corrosion or breathing damage). In such situations the patterns generation based on real tests or laboratory experiments is required. However, the limited number of models, their cost and the length of time needed mean that the number of patterns generated in this way is usually very low and may not be sufficient for a diagnosis to be properly established.

Fortunately an *unsupervised learning* technique may be applied for damage detection which uses *novelty detection* methods [1,4]. The principle governing

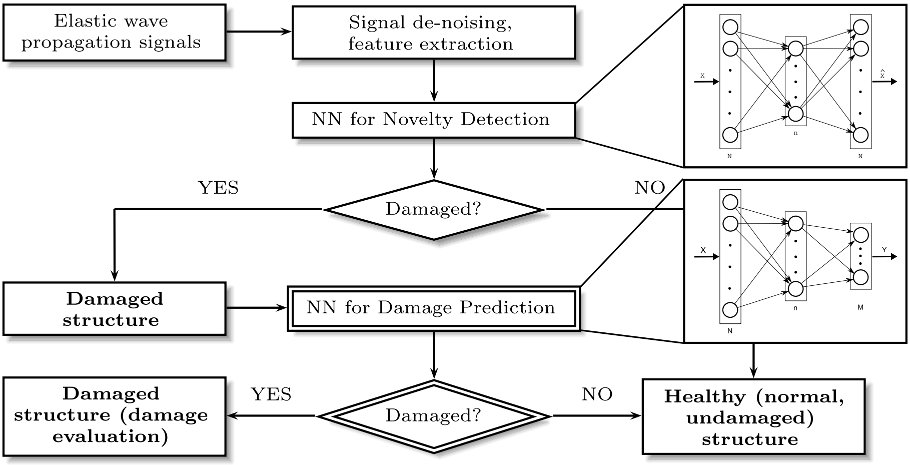

Elastic wave propagation signals
Signal de-noising, feature extraction
NN for Novelty Detection
x
N
n
N
x̂
YES
Damaged?
NO
x
N
n
M
Y
Damaged structure
NN for Damage Prediction
Damaged structure (damage evaluation)
YES
Damaged?
NO
Healthy (normal, undamaged) structure

Fig. 1. Flow diagram illustrating the system for damage detection and evaluation

novelty detection is that only training data from the normal operating condition of the structure is used to establish the diagnostics. A model of the normal condition is created and newly acquired data is compared with that of the model. If there are any significant deviations the algorithm indicates novelty. This means that the system has departed from the normal condition and may be damaged. If the training data is generated from a model, only the undamaged condition is required and it will simplify numerical computations. From an experimental point of view there is no need to damage the structure being investigated.

A diagnosis system flow diagram and possible applications of NNs are shown in Fig. 1. All the steps in the system are briefly discussed in the next subsections.

2.1 Elastic Waves

One very promising nondestructive technique that is suitable for the SHM systems under discussion utilizes the phenomenon of elastic wave propagation in solids [5]. For wave excitation and sensing the structure is usually equipped with piezoelectric transducers which can be used as waves actuators and sensors. Alternatively, non-contact methods like laser vibrometry can be used for elastic wave sensing and their multidimensional analysis [6].

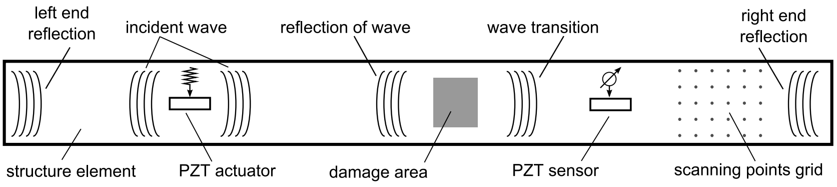


Fig. 2. Idea of elastic waves utilization

The principle governing the use of the elastic wave propagation phenomenon (Fig. 2) rests on the assumption that any obstacles will cause wave reflections and affect theirs transition through a damaged area. The signals received can be subjected to a procedure which pinpoints the location of the disturbance and predicts its nature and extent.

2.2 Signal Processing

The first stage of the SHM system, beyond sensor level, is signal cleansing. It was found that both environmental conditions and equipment precision may significantly affect the elastic wave signals measured. It is usually related to sensor and cable noise, structure's ambient vibration, environmental effects, incidental events, etc. Fortunately, thanks to a wide range of data processing techniques, the influence of signal noise can be decreased.

In the algorithm proposed the application of 1-D wavelet denoising reduced signals' digital nature, while a high-pass Chebyshev filter attenuated low frequency bins. Then feature extraction was achieved by determining various wave parameters such as amplitude, spectral density, correlation factor, etc. Alternatively a statistical algorithm of Principal Component Analysis (PCA) [7] was used in order to reduce signal dimension. As a result signal length was decreased from 2501 values to 16 principal components. This number of components ensured that reconstruction of the elastic wave signals could be performed with 99.5% accuracy.

2.3 Novelty Detection

The idea upon which novelty detection is based is that only training data obtained from the normal operating condition of the structure is used to establish the diagnostics [1,4]. First a model in normal condition is created and then newly acquired data is compared with that of the model. If there are any significant deviations, the algorithm indicates novelty. It means that the system has departed from its normal condition and may be damaged. This procedure is recognized as a damage detection algorithm and gives a mainly qualitative indication that damage may be present in the structure.

Here, for the purpose of novelty detection, autoassociative NNs were applied and learned using the Levenberg-Marquardt algorithm. When the trained network is fed with inputs obtained from a damaged state of the system, the novelty index $NI(\boldsymbol{x}) = \|\boldsymbol{x} - \bar{\boldsymbol{x}}\|$ (defined as the Euclidean distance between the target outputs $\boldsymbol{x}$ and the NN outputs $\bar{\boldsymbol{x}}$) will increase [4]. If the learning was successful, the index will satisfy $NI(\boldsymbol{x}) \approx 0$ for data obtained from the undamaged state. However, if data is obtained from a damaged structure, the novelty index will indicate an abnormal condition providing a non-zero value.

Although in general the novelty detection provides a two level classification (damaged, undamaged), it can be considered also as a multilevel classification indicating e.g. damage type, severity or number of damages. Obviously improved classification accuracy involves the application of a NN designed for multilevel classification. However, such an approach requires prior knowledge of the possible damage scenarios since the algorithm is based on class labels assigned to samples of measured data.

2.4 Damage Evaluation

Damage evaluation is often related to a regression problem and provides an approximation of the extent of damage, its location, etc. In the procedure proposed, feedforward NNs were used to predict the extent of damage in laboratory models of strip elements.

Based on the laboratory tests performed a damage pattern database, corresponding to both healthy and damaged structures, was defined. Then the NN architecture was designed and customized by minimizing a test Mean Square Error (MSE). The end results were preceded by many repetitions of NN training with various input vectors related to the extracted wave parameters, signal type (continuous or impulse sine wave – CSW or ISW), frequencies, sources, etc.

Unfortunately a number of patterns obtained from studied specimens is relatively small and the diagnostic had been established using only 26 patterns corresponding to one damage scenario. Due to this fact the number of NN parameters should be lower than the number of patterns used for learning.

3 Tests on Damage Identification

Let us consider a specimen steel strip (808x32x2 mm) where damages were introduced by notching across the width of the specimen for 20 mm from an outer edge to the opposite one. Two piezoelectric transducers served as actuators and sensors of elastic waves. Time signals measured at the actuator position for selected extents of damage are shown in Fig. 3. Unfortunately the disturbances introduced by damage are visually undetectable so that all three graphs in Fig. 3 appear identical. In addition most of the procedures described in the literature refer to analysis of much simpler signals [5,6].

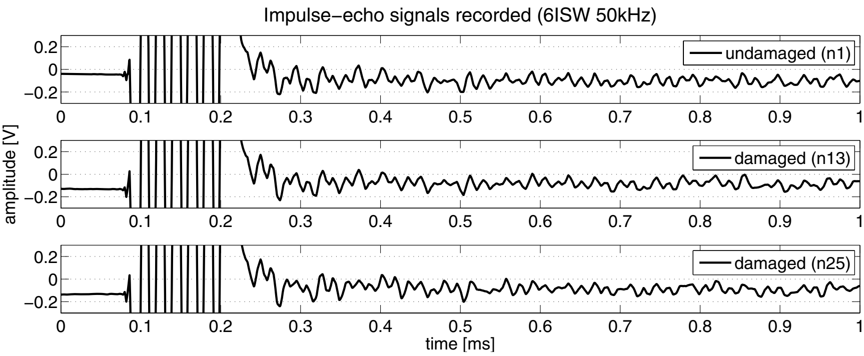
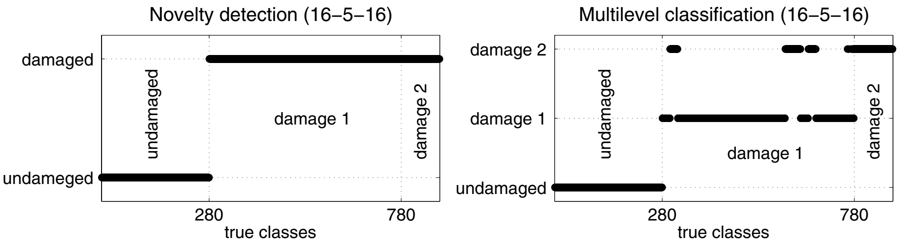


Fig. 3. Time signals recorded in steel strip (6 impulse sine waves 50 kHz)

As the measured signals were quite complex, advanced signal processing techniques were used for data cleansing and computing parameters for further analysis, chiefly wavelet denoising, data filtering, Fourier transformation and PCA. Next, the damage database obtained was used for training the designed diagnosis system to detect damage and evaluate its extent. It is worth mentioning that a huge computational effort was made to improve the training procedure with particular attention paid to various definitions of the input vectors, tuning the architecture of the NNs, parameter adjustment and repetitions of the learning algorithm, etc. The main objective of this was to decrease damage identification error and improve the generalization ability of the system.

3.1 Training Novelty Detection

The first identification level of the proposed diagnosis system was novelty detection. Autoassociative NNs and a novelty index were used for this purpose. From the 31 patterns obtained from laboratory tests the following specimen classes were assigned: undamaged and damaged (one or two damages).



(a) Novelty detection.

(b) Multilevel classification.

Fig. 4. Results of testing pattern classification using NN (16-5-16) and novelty index

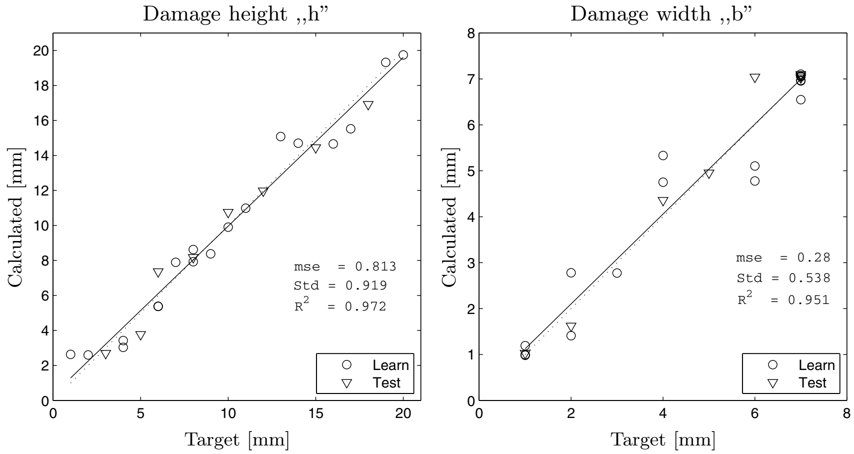


Fig. 5. Results of training damage evaluation (1-3-2): damage height and width

Since the number of patterns was relatively low it was increased by adding to the source signals various time histories of measurement noise. In such a way 1120 patterns corresponding to the undamaged case were used for learning the autoassociative NN while a further 880 patterns were used for NN testing. It may be seen in Fig. 4(a) that, through the application of PCA and NNs, extremely accurate pattern classification was achieved, even for patterns relating to incipient damage. It was proved that the NN with architecture 16-5-16 could correctly separate the classes of damaged and undamaged structures for all the damage scenarios considered. Next, a novelty index was used for multilevel classification and separation of particular structure classes (undamaged, one damage, two damages). The results obtained are shown in Fig. 4(b). The biggest classification errors were associated with the separation of inter-classes. Novelty index accuracy in this case was approximately 90% for one damage, whereas the undamaged and the 2 damage cases were classified with excellent (100%) accuracy.

3.2 Training Damage Evaluation

In the case when the diagnosis system has warned that an anomaly has been detected, another type of NN can be applied to evaluate the damage parameters. Since the position of any damage was considered constant, the only parameter which it was necessary to identify here was damage extent (height and width).

The input vectors consisted mainly of spectrum magnitudes and threshold amplitudes were defined. Damage prediction was established based on 26 patterns related to an undamaged structure and to a structure with one area of increasing damage extent. After several repetitions of NN training, the best results obtained at the present stage of the performed tests are shown in Fig. 5 for both learning (circle) and testing (triangle) damage evaluation. It can be seen from these plots that the testing accuracy of damage extent varied between 0.10 and 1.13 mm. In comparison with the other results relating to the wide range of signals studied and their parameters, the accuracy obtained was good and remained at a similar level (Table 1). Taking into account the mean accuracy obtained (computed for all testing patterns) it emerges that the most suitable data, among all the experiments performed, were impulse signals consisting of 4 and 6 sine waves and theirs dynamic amplitudes A_d or magnitudes M.

Table 1. Error values of predicted damage extent in case of the NNs testing

Signal	Input	Damage width error [mm]			Damage height error [mm]		
type	vector	mean	min	max	mean	min	max
CSW 38 kHz	$A_{f\,(1\times N)}$	0.37	0.003	1.35	0.57	0.09	1.26
CSW 18-38-44 kHz	$A_{d\,(3\times N)}$	0.21	0.007	0.48	0.51	0.17	0.99
ISWx1 38 kHz	$A_8A_{9\,(2\times N)}$	0.28	0.002	0.85	0.52	0.02	1.66
ISWx4 42 kHz	$M_{1\,(1\times N)}$	**0.12**	0.011	0.55	0.51	0.10	1.13
ISWx6 38 kHz	$A_{d\,1\mathrm{to}3\,(3\times N)}$	0.22	0.001	0.88	**0.26**	0.03	0.55

4 Conclusions and Final Remarks

The application of NNs in the field of structural test and health monitoring was investigated. They were used here as a tool of automatic signal analysis of elastic waves. The case study concerned laboratory specimens of damaged and undamaged steel strips. Classical analysis of the signals measured appeared impossible due to limitations such as the relatively short model length, damage located close to the site of excitation, superposition of many reflections, operating constraints of the piezoelectric transducers and testing equipment. However NNs have been shown to be useful for the analysis of such *complex* signals. The procedure presented here allows automation of structure testing and can be used for on-line SHM systems.

The number of damage patterns used for novelty detection was increased here by adding measurement noise to the measured signals. Although the training data was noised the trained NNs retained the same accuracy of prediction. It proves the generalization ability of a trained NN.

Further work in the field of SHM should be focused on system validation during laboratory tests performed on real structures or their elements. It is also worth studying the application of other soft computing methods, such as Support Vector Machines or Bayesian NN, in order to improve the accuracy of both pattern classification and prediction of damage parameters.

Acknowledgments. The Polish Ministry of Science and Higher Education is gratefully acknowledged for its financial support to the research activity presented: Grants Nos. N N501 134336 and R10 005 02.

References

1. Worden, K., Dulieu-Barton, J.M.: An Overview of Intelligent Fault Detection in Systems and Structures. Structural Health Monitoring 3, 85–98 (2004)
2. Staszewski, W., Boller, C., Tomlinson, G.: Health Monitoring of Aerospace Structures: Smart Sensor Technologies and Signal Processing. John Wiley & Sons, Chichester (2004)
3. Waszczyszyn, Z., Ziemiański, L.: Neural networks in the identification analysis of structural mechanics problems. In: Mroz, Z., Stavroulakis, G.E. (eds.) Parameter Identification of Materials and Structures. CISM Courses and Lectures, vol. 469, pp. 259–340. Springer, Heidelberg (2005)
4. Hernandez-Garcia, M.R., Sanchez-Silva, M.: Learning Machines for Structural Damage Detection. In: Lagaros, N.D., Tsompanakis, Y. (eds.) Intelligent Computational Paradigms in Earthquake Engineering, pp. 158–187. Idea Group Publishing (2007)
5. Lee, B.C., Staszewski, W.: Lamb wave propagation modeling for damage detection: II. Damage monitoring strategy. Smart Mater. Struct. 16, 260–274 (2007)
6. Staszewski, W., Lee, B.C., Mallet, L., Scarpa, F.: Structural health monitoring using laser vibrometry: I. Lamb wave sensing. Smart Mater. Struct. 13, 251–260 (2004)
7. Kuźniar, K., Waszczyszyn, Z.: Neural Networks and Principal Component Analysis for Identification of Building Natural Periods. J. of Comp. in Civil Engineering 20, 431–436 (2006)

Persistent Activation Blobs in Spiking Neural Networks with Mexican Hat Connectivity

Filip Piekniewski[⋆]

Faculty of Mathematics and Computer Science,
Nicolaus Copernicus University, 87-100 Torun, Poland
philip@mat.umk.pl
http://www.mat.umk.pl/~philip/

Abstract. Short range excitation, long range inhibition sometimes referred to as mexican hat connectivity seems to play important role in organization of the cortex, leading to fairly well delineated sites of activation. In this paper we study a computational model of a grid filled with rather simple spiking neurons with mexican hat connectivity. The simulation shows, that when stimulated with small amount of random noise, the model results in a stable activated state in which the spikes are organized into persistent blobs of activity. Furthermore, these blobs exhibit significant lifetime, and stable movement across the domain. We analyze lifetimes and trajectories of the spots, arguing that they can be interpreted as basic computational *charge* units of the so called *spike flow model* introduced in earlier work.

Keywords: spiking networks, mexican hat connectivity, spike flow model.

1 Introduction

It is a subject of ongoing discussion of what is the elementary computational unit of the brain, whether this important role should be attributed to a neuron and an action potential, or rather a group of neurons, possibly a polychronous group of spikes [1] or maybe some bigger ensemble with more complex dynamics (microcolumn etc.). In our previous work [2,3,4,5] we studied the so called *spike flow model* in which neuron-like units exchange quants of some persistent charge (that is conserved by the dynamics), leading eventually to a winner-take-all dynamics and scale-free (power law) connectivity of the resulting charge transfer graph. The study was motivated by experimental results, which showed that functional brain networks have certain connectivity properties [6,7]. The model, though leading to similar properties of the charge exchange graph (power law degree distribution with exponent $\gamma = 2$), lacked exact biological interpretation in terms of single neurons since single neuronal spikes are not persistent and furthermore a single neuron cannot hold obtained spikes for later. Such a property

[⋆] This work is supported by the Polish Minister of Scientific Research and Higher Education grant N N201 385234 (2008-2010).

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 64–71, 2010.
© Springer-Verlag Berlin Heidelberg 2010

could be however attributed to larger ensembles of neurons, particularly those having recurrent connections. In such case the local excitation can be preserved via loopback connections (in a way, stored). In this paper we show, that a grid of spiking neurons with short range excitation, long range inhibition stimulated via small amount of random noise converges to a homeostatic state, in which well delineated, persistent activity blobs emerge. Furthermore these blobs are able to travel significant distances across the domain. The total number of blobs is preserved (though some blobs vanish, and new are born) resembling the charge flow of the *spike flow model*.

2 The Model

The model consists of a grid of Eugene M. Izhikevich phenomenological simple neurons [8,9,10] governed by

$$\begin{cases} v' = & 0.04v^2 + 5v + 140 - u + I \\ u' = & a(bv - u) \end{cases} \tag{1}$$

the parameters are set as in the sample Matlab program in section IV of [8]. The connectivity is shown on figure 1. Neurons are organized into grid (torus topology), connected locally with mexican hat like weights computed from $50 \cdot (e^{-d^2} - 0.5 \cdot e^{-0.5 \cdot d^2})$ where $d = 0.5\sqrt{dx^2 + dy^2}$ is the grid distance (grid step is assumed to be 1). The inhibitory to excitatory weight was -20, while the excitatory to inhibitory weight was $\frac{10}{\#\text{ neurons}}$. We neglect conduction delays, since the connectivity is by definition local. The program was implemented in Matlab[1] in a fashion similar to that of original E. Izhikevich script. Various sizes

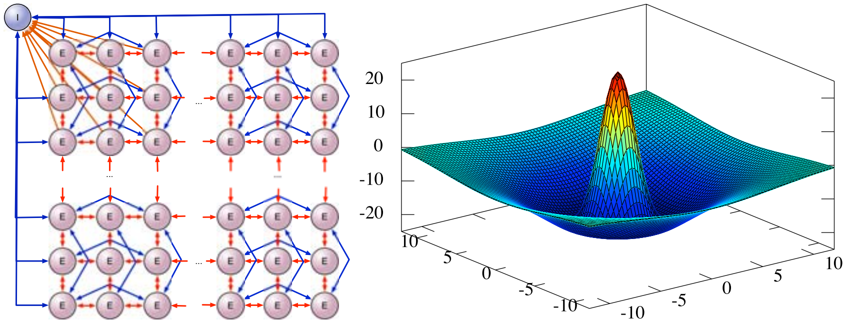


Fig. 1. The model consists of a grid of excitatory neurons connected to one inhibitory neuron, which has feedback inhibitory connections. Excitatory neurons excite their nearest neighbors but inhibit those at larger distances as $50 \cdot (e^{-d^2} - 0.5 \cdot e^{-0.5 \cdot d^2})$ (left) where $d = 0.5\sqrt{dx^2 + dy^2}$. The grid step is assumed to be 1.

[1] The script is available http://www.mat.umk.pl/~philip/papers/MH_grid.m

of the model were simulated, here we present figures obtained from 50x50 (1s - 1000 steps) and 80x80 (5s - 5000 steps) simulations.

In order to analyze the behavior of the blobs a following methodology was used:

- The voltage field was thresholded at fairly high value $v = -30$
- The resulting bitmap was cleaned with Matlabs `bwmorph`[2] function (single isolated pixels were removed)
- The bitmap was decomposed with Matlabs `bwlabel` funtion into individual blobs.
- Each blob was attributed a blob center in its center of mass.
- The list of existing blobs was looked up in search for nearest blobs to those found in current iteration.
- If previously existing blob was found within the distance of 3 units from any newly found, then the new one was resolved as the existing one that must have moved from previous time step. In that case the center of the existing blob gets updated (simple movement tracking).
- If there were no previously existing blobs within the range of 3 units, the blob was pronounced a new.
- After that step if some blob was neither updated nor created, it was removed from the list.

3 The Spike Flow Model

The spike flow model has been introduced in [2,3] to comprehend with scale-free connectivity that has been found in some functional fMRI based networks [6,7] (none of the preexisting scale-free networks model was suitable to explain these phenomena).

The model consists of nodes σ_i, $i = 1 \ldots N$. Each node's state is described by a natural number from some fixed interval $[0, M_i]$. The network is built on a complete graph in that there is a connection between each pair of neurons $\sigma_i, \sigma_j,\ i \neq j$, carrying a real-valued weight $w_{ij} \in \mathbb{R}$ satisfying the usual symmetry condition $w_{ij} = w_{ji}$, moreover $w_{ii} := 0$. The values of w_{ij} are drawn independently from the standard Gaussian distribution $\mathcal{N}(0,1)$ and are assumed to remain fixed in the course of the network dynamics. The model is equipped with the Hamiltonian of the form:

$$\mathcal{H}(\bar{\sigma}) := \frac{1}{2} \sum_{i \neq j} w_{ij} |\sigma_i - \sigma_j| \tag{2}$$

if $0 \leq \sigma_i \leq M_i,\ i = 1, \ldots, N$, and $\mathcal{H}(\bar{\sigma}) = +\infty$ in the other case. Here $\bar{\sigma}$ denotes of the state of the whole system. The dynamics of the network is defined as follows: at each step we randomly choose a pair of "neurons" (units) $(\sigma_i, \sigma_j),\ i \neq j$, and denote by $\bar{\sigma}^*$ the network configuration resulting from the original configuration $\bar{\sigma}$ by decreasing σ_i by one and increasing σ_j by one, that is to say by

[2] `bwmorph` and `bwlabel` are supplied with the image processing toolbox.

letting a unit charge transfer from σ_i to σ_j, whenever $\sigma_i > 0$ and $\sigma_j < M_j$. Next, if $\mathcal{H}(\bar{\sigma}^*) \leq \mathcal{H}(\bar{\sigma})$ we accept $\bar{\sigma}^*$ as the new configuration of the network whereas if $\mathcal{H}(\bar{\sigma}*) > \mathcal{H}(\bar{\sigma})$ we accept the new configuration $\bar{\sigma}^*$ with probability $\exp(-\beta[\mathcal{H}(\bar{\sigma}^*) - \mathcal{H}(\bar{\sigma})])$, $\beta > 0$, and reject it keeping the original configuration $\bar{\sigma}$ otherwise, with $\beta > 0$ standing for an extra parameter of the dynamics, in the sequel referred to as the inverse temperature conforming to the usual language of statistical mechanics.

The model results in a scale-free charge transfer graph with exponent $\gamma = 2$ (in agreement with empirical data), where the weight of each edge corresponds to the frequency of charge exchange events that were conducted along that edge (see [3] for details). The weak point of that model (though quite universal from mathematical point of view) is that it did not have an exact interpretation in terms of neurobiology, since it is not clear what the computational units and charge quants correspond to. The present paper is aimed to provide a direct[3] link between biologically feasible spiking networks and the *spike flow model*.

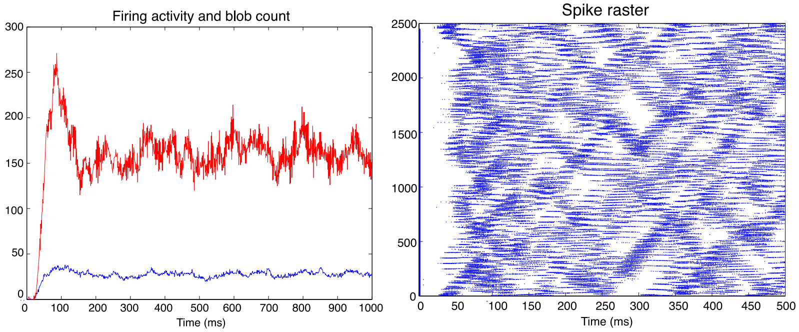


Fig. 2. Firing activity and the emergence of blobs on 50x50 domain. It takes some time before the model ignites the homeostatic blob dynamics (and it depends on the initial conditions). Nevertheless once the blobs emerge, they are persistent and firing activity levels off at some medium magnitude. The right figure shows the spike raster.

4 Results

At the beginning of the simulation the system fires single spikes in a unorganized manner, reflecting the random stimulation. At some point more spikes appear to synchronize. Eventually a number of activity blobs emerge and start to move across the domain. Soon after the initial activity jump (see figure 2 left), the system levels off in a homeostatic state in which the blobs are persistently emerging and moving (see figure 3).

The rate at which the system arrives at the homeostatic regime depends somewhat on the initial conditions. It seems that the system requires some time to

[3] There are also other possible spike flow model interpretations, in terms of times spent by units in a certain dynamic attractor and so on. In this paper however, we show a rather straightforward interpretation.

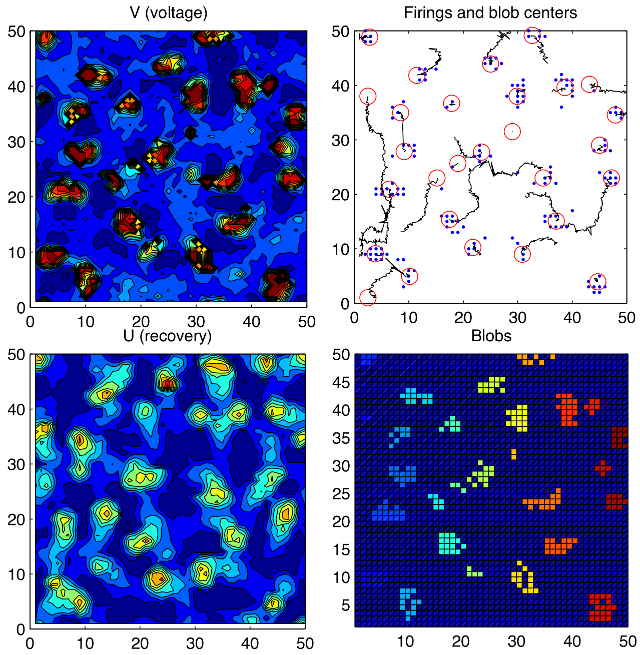


Fig. 3. 50x50 neuron domain. The voltage v is plotted on the top left subfigure. Bottom left shows the recovery variable u. Bottom right plot shows the thresholded voltage value divided into individual blobs. Each blob's center of a mass is shown with a circle on top right subfigure. The dots are spikes, the black lines are trajectories left by each blob as it moves. A realtime movie of the simulation of 80x80 domain is available `http://www.mat.umk.pl/~philip/ICAISC2009/80_realtime.mov` , and slowed down 8 times `http://www.mat.umk.pl/~philip/ICAISC2009/80_8xslower.mov`.

synchronize. Artificially firing all neurons at the first step leads to faster convergence[4]. Nevertheless once the blobs emerge, they stay forever[5], and so the initial conditions don't seem to be very important for blob features. An important question addressed in this paper is whether the blobs satisfy the conditions which allow them to be considered as the charge packets exchanged in the spike flow model, that is:

– Is the average blob lifetime long?
– Do blobs manage to travel long distances?
– Are blobs similar in sizes and activations they carry?

These questions will be answered in paragraphs below.

[4] Artificial firing of all neurons is the initial condition used in this paper.

[5] Existence of blobs is very stable in the model with local mexican hat connectivity (presented here). However addition of random edges may introduce time periodicity (caused by desynchronization) and cause blobs to vanish and reappear and so on.

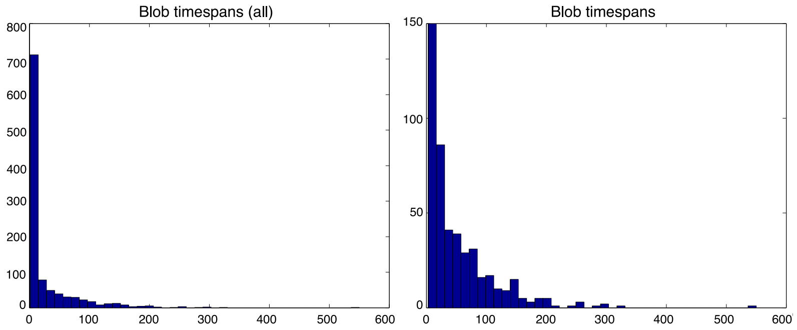


Fig. 4. The distribution of timespans among the blobs on 50x50 domain. As previously the left figure shows all timespans, while the right one only those which persisted for more than 2ms.

Lifespan. The blobs are rather persistent. As seen in figure 4 for the 50x50 domain some blobs are able to survive nearly 600ms. The distribution however is concentrated on short living blobs. This is due to the properties of blob finding algorithm - many blobs are only pinpointed in a single time step. This happens frequently, whenever two or three random spikes appear nearby. Such random fluctuations do not give a rise to a "real" blob, but instead mess the statistics with false positives. Since due to short lifetime these false positive blobs do not move, we compute some statistics after throwing away blobs living less than some threshold (2-10 time steps). Median timespan of "true" blobs is between 40-70ms (depending on the size of the domain and time of simulation). The oldest blobs at 80x80 domain arrive near 1s life, which is exactly the timescale expected with the spike flow model.

Distances. While alive, blobs move across the domain. It turns out their movement is not like a random walk in which they would constantly change the

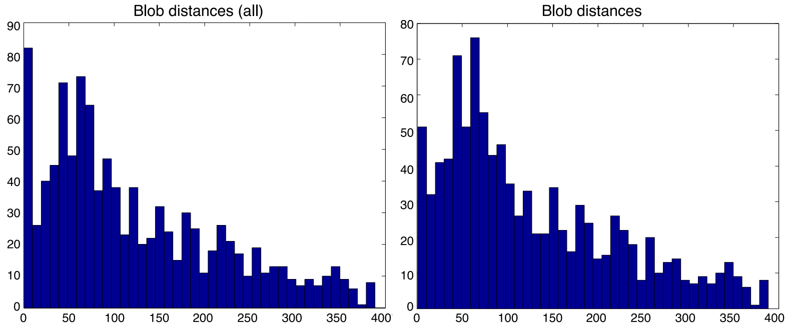


Fig. 5. The distribution of distances traveled by blobs on 50x50 domain during 4000ms simulation. The left figure shows all distances including those, traveled by flase positive blobs (which survive only one time step), the right one shows only those which persisted at least 2 ms.

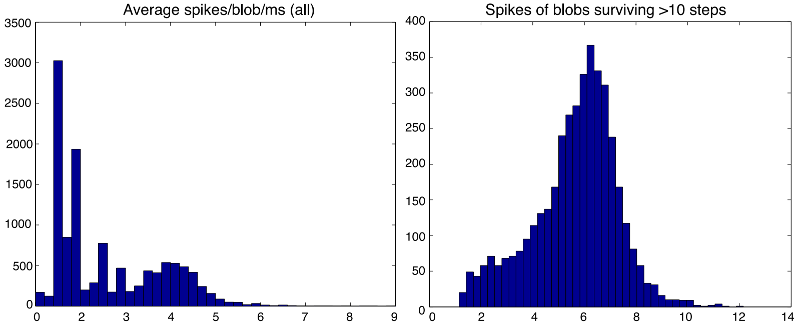


Fig. 6. The distribution of an average number of spikes firing within the blob per time step. After neglecting false positives (blobs that only emerged for very short time, in this case < 10) one obtains a fairly centered distribution. This shows that all persistent blobs are rather equally active.

direction of movement and eventually drifted in brownian manner. Instead it seems like the blobs have true velocities which are changed rarely (see sample trajectories on figure 3 top right). The best blobs managed to travel nearly 400 distance units on 50x50 domain. With 80x80 domain some blobs traveled nearly 2000 units (with median near 100)! Furthermore these statistics can be diminished by the methodology - since the domain is a torus some blobs disappear at one side and reappear at the other. With the current algorithm such spots are treated separately, whereas in fact they are the same blob. Nevertheless, even with the imperfect statistics the conclusion is that blobs manage to travel significant distances.

Activities. The figure 3 (bottom right) might suggest that the blobs are of various sizes and shapes. This is a rather false conviction, since the spots constantly change, at one frame the blob appears as large while a few steps later it is very small. To obtain a trustworthy statistics about the blobs activity a following method was used:

- Whenever a neuron spiked, an algorithm looked for a blob within a distance of 6 units from the spike
- If it found one, the spike was attributed as coming from that particular blob. In other case it was ignored.
- When the spot was at the end of its life, the total number of spikes it collected was divided by its lifetime, to obtain average spiking activity. These averages were saved to obtain a histogram in figure 6.

The results are shown in figure 6. Again the statistics are corrupted with false positive transient blobs. After throwing away any spot that survived less that 10 steps, one obtains a fairly well centered distribution with a majority of blobs having 5-7 spikes/ms. This shows that in fact most of the blobs are much alike, and carry the same amount of "activity".

5 Conclusions

The presented model is aimed at bridging biologically plausible dynamical spiking neural networks with the *spike flow model* which itself is aimed at showing where the scale-free connectivity in functional brain networks might come from. The persistent activity blobs described here seem to be right candidates for the charge units that are being exchanged in the spike flow model. As shown by simulations, the blobs are persistent, travel significant distances, operate on the right timescale and on average carry the same amount of ”activity”. The ”neurons” of the *spike flow model* can in this context be interpreted as subsets of the domain. In this simple case the domain in 2d (resembling the cortex), but such blobs should also appear with higher dimensionality. In particular the long range myelinated cortico-cortical connections can form wormholes that teleport a blob from one cortical area to another, giving them more freedom (the spike flow model in the original setup is a mean field model, but many of its properties remain valid when it is submerged in rich enough topology).

Acknowledgments. The author gratefully acknowledges the support from the Polish Minister of Scientific Research and Higher Education grant N N201 385234 (2008-2010). The author also appreciates fruitful collaboration with Dr Tomasz Schreiber.

References

1. Izhikevich, E.M.: Polychronization: Computation with spikes. Neural Comput. 18(2), 245–282 (2006)
2. Pię kniewski, F., Schreiber, T.: Emergence of scale-free spike flow graphs in recurrent neural networks. In: Proc. IEEE FOCI 2007, pp. 357–362 (2007)
3. Pię kniewski, F., Schreiber, T.: Spontaneous scale-free structure of spike flow graphs in recurrent neural networks. Neural Networks 21(10), 1530–1536 (2008)
4. Pię kniewski, F.: Emergence of scale-free graphs in dynamical spiking neural networks. In: Proc. IEEE International Joint Conference on Neural Networks, Orlando, Florida, USA, August 2007, pp. 755–759. IEEE Press, Los Alamitos (2007)
5. Piekniewski, F.: Robustness of power laws in degree distributions for spiking neural networks. In: Proc. IEEE 2009 International Joint Conference on Neural Networks (IJCNN), Atlanta, Georgia, USA, June 2009, pp. 2541–2546. IEEE Press, Los Alamitos (2009)
6. Sporns, O., Chialvo, D.R., Kaiser, M., Hilgetag, C.C.: Organization, development and function of complex brain networks. Trends Cogn. Sci. 8(9), 418–425 (2004)
7. Eguíluz, V.M., Chialvo, D.R., Cecchi, G.A., Baliki, M., Apkarian, A.V.: Scale-free brain functional networks. Phys. Rev. Lett. 94(1) (January 2005)
8. Izhikevich, E.M.: Simple model of spiking neurons. IEEE Transactions on Neural Networks (14), 1569–1572 (2003)
9. Izhikevich, E.M.: Which model to use for cortical spiking neurons? IEEE Transactions on Neural Networks 15(5), 1063–1070 (2004)
10. Izhikevich, E.M.: Dynamical systems in Neuroscience: The Geometry of Excitability and Bursting. MIT Press, Boston (2006)

Neurogenetic Approach for Solving Dynamic Programming Problems

Matheus Giovanni Pires and Ivan Nunes da Silva

State University of Feira de Santana, Department of Computer Engineering,
CEP 44031-460, Feira de Santana, BA, Brazil
University of São Paulo, Department of Electrical Engineering,
CP 359, CEP 13566.590, São Carlos, SP, Brazil
mgpires@gmail.com, insilva@sc.usp.br
http://laips.sel.eesc.usp.br

Abstract. Dynamic programming has provided a powerful approach to solve optimization problems, but its applicability has sometimes been limited because of the high computational effort required by the conventional algorithms. This paper presents an association between Hopfield networks and genetic algorithms, which cover extensive search spaces and guarantee the convergence of the system to the equilibrium points that represent feasible solutions for dynamic programming problems.

Keywords: Dynamic programming, genetic algorithms, Hopfield network.

1 Introduction

The parallel nature of artificial neural networks makes them suitable for solving several classes of optimization problems, such as combinatorial [1], linear programming [2], nonlinear optimization [3] and dynamic programming [4].

Usually, the solution of optimization problems by dynamic programming involves the recurrence relations developed by Bellman [5]. Although the dynamic programming has been used for solving several classes of optimization problems, it has computational inefficiency (e.g. CPU time, memory), so the neural networks become an interesting approach that can be applied in these problems.

Another approach that has been applied to optimization problems and has shown promise for solving such problems efficiently is Genetic Algorithm (GA). In this paper, we perform an analysis of a neurogenetic architecture, not depending on weighting and/or penalty parameters, for dynamic programming.

2 Shortest Path Problem

A typical dynamic programming problem can be modeled as a set of source and destination nodes with n intermediate stages, m states in each stage, and metric data $d_{xi,(x+1)j}$, where x is the index of the stages, and i and j are the indices of the states in each stage, as shown in Fig. 1.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 72–79, 2010.
© Springer-Verlag Berlin Heidelberg 2010

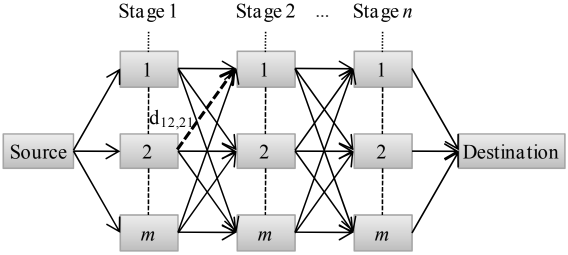


Fig. 1. Shortest path problem

The goal of the dynamic programming considered here is to find a valid path, which starts at the source node, visits one and only one state node in each stage, reaches the destination node, and has a minimum total length (cost).

3 The NeuroGenetic Approach (NGA)

As introduced in [1], the node equation for the continuous-time network with N-neurons is given by:

$$\dot{u}_i(t) = -\eta \cdot u_i(t) + \sum_{j=1}^{N} T_{ij} \cdot v_j(t) + i_i^b \tag{1}$$

$$v_i(t) = g(u_i(t)) \tag{2}$$

where $u_i(t)$ is the current state of the i-th neuron, $v_j(t)$ is the output of the j-th neuron, i_i^b is the offset bias of the i-th neuron, $\eta \cdot u_i(t)$ is a passive decay term, and T_{ij} is the weight connecting the j-th neuron to i-th neuron. It is shown in [1] that the network equilibrium points correspond to values $v(t)$ for which the energy function (3) associated with the network is minimized:

$$E(t) = -\frac{1}{2} v(t)^T \cdot T \cdot v(t) - v(t)^T \cdot i^b \tag{3}$$

The mapping of programming problems using a Hopfield network consists of determining the weight matrix T and the bias vector i^b to compute equilibrium points. A modified energy function $E^m(t)$ is used here, which is defined by:

$$E^m(t) = E^{conf}(t) + E^{op}(t) \tag{4}$$

where $E^{conf}(t)$ is a confinement term that groups all structural constraints associated with the problem, and $E^{op}(t)$ is an optimization term that conducts the network output to the equilibrium points corresponding to a cost constraint. Thus, the minimization of $E^m(t)$ is conducted in two stages:

i) Minimization of the Term $E^{conf}(t)$:

$$E^{conf}(t) = -\frac{1}{2}v(t)^T \cdot T^{conf} \cdot v(t) - v(t)^T \cdot i^{conf} \tag{5}$$

where: $v(t)$ is the network output, T^{conf} is weight matrix and i^{conf} is bias vector belonging to E^{conf}. This corresponds to confinement of $v(t)$ into a valid subspace generated from structural constraints imposed by the problem. An investigation associating the equilibrium points with respect to the eigenvalues and eigenvectors of the matrix T^{conf} shows that all feasible solutions can be grouped in a unique subspace of solutions [4,6]. As consequence of the application of this approach, which is named by valid-subspace method, a unique energy term can be used to represent all constraints associated with the optimization.

ii) Minimization of the Term $E^{op}(t)$:

$$E^{op}(t) = -\frac{1}{2}v(t)^T \cdot T^{op} \cdot v(t) - v(t)^T \cdot i^{op} \tag{6}$$

After confinement of all feasible solutions to the valid subspace, a GA is applied to optimize $E^{op}(t)$ by inserting the values $v(t)$ into the chromosomes population. Thus, the operation of this hybrid system consists of three steps:

Step (I): Minimization of E^{conf}, corresponding to the projection of $v(t)$ in the valid subspace defined by:

$$v(t+1) = T^{val} \cdot v(t) + s \Longleftrightarrow v \leftarrow v = T^{val} \cdot v + s \tag{7}$$

where: T^{val} is a projection matrix ($T^{val} \cdot T^{val} = T^{val}$) and the vector s is orthogonal to this subspace ($T^{val} \cdot s = 0$). This operation corresponds to an indirect minimization process [4] of $E^{conf}(t)$, i.e. $T^{conf} = T^{val}$ and $i^{conf} = s$.

Step (II): Application of a *symmetric-ramp* activation function constraining $v(t)$ in a hypercube, i.e.

$$g(v_i(t)) = \begin{cases} 1 & \text{, if} \quad v_i(t) > 1 \\ v_i(t) & \text{, if} \quad 0 \le v_i(t) \le 1 \\ 0 & \text{, if} \quad v_i(t) < 0 \end{cases} \text{, where } v_i(t) \in [0,1] \tag{8}$$

Step (III): Minimization of E^{op}, which involves the application of a genetic algorithm to move $v(t)$ towards an optimal solution that corresponds to network equilibrium points, which are the solutions for the optimization problem.

As seen in Fig. 2, each iteration represented by the above steps has two distinct stages. First, as described in Step (III), v is updated using the GA. Second, after each updating given in Step (III), v is projected directly in the valid subspace by the modified Hopfield network. This second stage is an iterative process, in which v is first orthogonally projected in the valid subspace by applying Step (I)

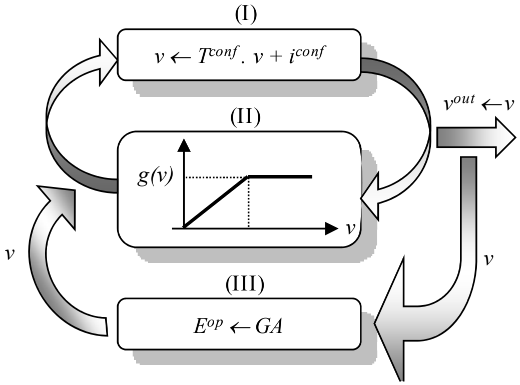


Fig. 2. The neurogenetic approach

and then thresholded in Step (II), so that its elements lie in the range defined by [0,1]. This process moves the network output to the equilibrium point that corresponds to the optimal solution for the optimization problem. The convergence is concluded when the values of v^{out} during two successive loops remain practically constant, where v^{out} in this case is equal to v.

4 Mapping of the Shortest Path Problem

Notation and Definitions

- The vector $p \in \Re^n$ represents the solution set of an optimization problem consisted of n nodes (neurons). Thus, the elements belonging to p have integer elements defined by:

$$p_i \in \{1, \ldots, n\}, \text{where } i \in \{1 \ldots n\} \tag{9}$$

  The vector p can be represented by a vector v, composed of ones and zeros, which represents the output of the network. In the notation using Kronecher products [7], we have:
- $\delta \in \Re^{n_x n}$ is a matrix defined by:

$$\delta_{ij} = \begin{cases} 1 \text{ , if } & i = j \\ 0 \text{ , if } & i \neq j \end{cases} \tag{10}$$

  $\delta(k) \in \Re^n$ is a column vector corresponding to k-th column of δ.
- $v(p)$ is an $n \cdot m$ dimensional vector representing the final form of the network output vector v, which corresponds to the problem solution denoted by p. The vector $v(p)$ is defined by:

$$v(p) = \begin{bmatrix} \delta(p_1) \\ \delta(p_2) \\ \vdots \\ \delta(p_n) \end{bmatrix} \tag{11}$$

- $vec(U)$ is a function which maps the $m_x n$ matrix to the $(n \cdot m)$-element vector v. This function is defined by:

$$v = vec(U) = \left[U_{11}U_{21} \cdots U_{m1}\ U_{12}U_{22} \cdots U_{m2}\ U_{1n}U_{2n} \cdots U_{mn} \right] \tag{12}$$

- $V(p)$ is an $n_x m$ dimensional matrix defined by:

$$V(p) = \begin{bmatrix} \delta(p_1)^T \\ \delta(p_2)^T \\ \vdots \\ \delta(p_n)^T \end{bmatrix} \tag{13}$$

  where $[V(p)]_{ij} = [\delta(p_i)]_j$.
- $P \otimes Q$ denotes the Kronecher product of two matrices. If P is an $n_x n$ matrix, and Q is an $m_x m$ matrix, then $P \otimes Q$ is an $(n \cdot m)_x (n \cdot m)$ matrix given by:

$$P \otimes Q = \begin{bmatrix} P_{11}Q & P_{12}Q & \cdots & P_{1n}Q \\ P_{21}Q & P_{22}Q & \cdots & P_{2n}Q \\ \vdots & \vdots & \ddots & \vdots \\ P_{n1}Q & P_{n2}Q & \cdots & P_{nn}Q \end{bmatrix} \tag{14}$$

- o^n and O^n are respectively the n-element vector and the $n_x n$ matrix of ones, i.e.

$$\left.\begin{matrix} [o]_i & = 1 \\ [O]_{ij} & = 1 \end{matrix}\right\} \text{for } i, j \in \{1, \ldots, n\} \tag{15}$$

- R^n is an $n_x n$ projection matrix, i.e. $R^n \cdot R^n = R^n$, which is given by:

$$R^n = I^n - \frac{1}{n} \cdot O^n \tag{16}$$

  The multiplication by R^n transforms the column sums of a matrix to zero.
- Another property of Kronecher products used here is the following one:

$$vec(Q \cdot V \cdot P^T) = (P \otimes Q) \cdot vec(V) \tag{17}$$

Defining Parameters for $E^{conf}(t)$ and $E^{op}(t)$

The equations of T^{val} and s are developed to force the validity of the structural constraints. These constraints, for dynamic programming problems, mean that one and only one state in each stage can be actived. Thus, the matrix $[V(p)]_{ij} \in \{1, 0\}$ is defined by:

$$\sum_{j=1}^{m} [V(p)]_{ij} = 1 \tag{18}$$

By using (7), a valid subspace ($V = T^{val} \cdot V + S$) for the dynamic programming problem can be represented by:

$$S = V = \frac{1}{m} o^n \cdot (o^m)^T \tag{19}$$

Equation (19) guarantees that the sum of the elements of each line of the matrix V takes values equal to 1. Therefore, the term $T^{val} \cdot V$ must also guarantee that the sum of the elements of each line of the matrix V takes value equal to zero. Using the properties of the matrix R^n, we have:

$$V \cdot R^m = T^{val} \cdot V \Longleftrightarrow I^n \cdot V \cdot R^m = T^{val} \cdot V \tag{20}$$

Using (19) and (20) in matrix equation of the valid subspace ($V = T^{val} \cdot V + S$),

$$V = I^n \cdot V \cdot R^m + \frac{1}{m} o^n \cdot (o^m)^T \tag{21}$$

Applying operator $vec(.)$, which is given by (17), in (21), we have:

$$vec(V) = (I^n \otimes R^m) \cdot vec(V) + \frac{1}{m}(o^n \otimes o^m) \tag{22}$$

Converting $vec(V)$ to v, the parameters T^{val} and s, which are belonging to valid subspace defined in (7), can now be extracted as follows:

$$v = \underbrace{(I^n \otimes R^m)}_{T^{val}} \cdot v + \underbrace{\frac{1}{m}(o^n \otimes o^m)}_{s} \tag{23}$$

The energy function E^{op} of the modified Hopfield network for the shortest path problem is projected to find a minimum path among all possible paths. When E^{op} is minimized, the optimal solution corresponds to the minimum energy state of the network. For this purpose, the function E^{op} is defined by:

$$E^{op} = \frac{1}{4}[\sum_{x=1}^{n-1}\sum_{i=1}^{m}\sum_{j=1}^{m} d_{xi,(x+1)j} \cdot v_{xi} \cdot v_{(x+1)j} + \sum_{x=2}^{n}\sum_{i=1}^{m}\sum_{j=1}^{m} d_{(x-1)j,xi} \cdot v_{xi} \cdot v_{(x-1)j}] +$$
$$+ \sum_{x=1}^{1}\sum_{i=1}^{m} d_{source,xi} \cdot v_{xi} + \sum_{x=n}^{n}\sum_{i=1}^{m} d_{xi,destination} \cdot v_{xi} \tag{24}$$

Therefore, optimization of E^{op} corresponds to minimize each term given by (24) in relation to v_{xi}.

5 Genetic Algorithm for Objective Function Optimization

In GA, potential solutions to a problem are represented as a population of chromosomes and each chromosome stands for a possible solution.

Codification: The size of the chromosomes is equal $n \cdot m$ and each chromosome is encoded as a vector of floating point numbers, in which each m components of the vector are the states of each stage, as illustrated in Fig. 3.

Initial Population: The population size used here was 100 individuals. The initial population is generated by introducing a chromosome that represents the

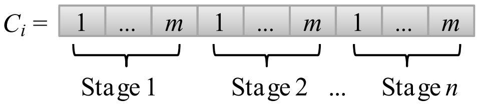


Fig. 3. Codification of chromosomes

values $v(t)$ previously obtained from Steps (I) and (II) described in Section 2. A total of 10% of the chromosomes are generated around $v(t)$, i.e., a random number between 0.001 and 0.5 is added or subtracted to $v(t)$. The remaining chromosomes are generated randomly.

Fitness Function: The fitness function evaluates each chromosome by verifying its environment adaptation. The fitness function to be here minimized is that given in (24) and the most adapted individual will have the minimum fitness value.

Intermediate Population: The selection method used here to separate the intermediate population was the N-Way Tournament selection [8]. The selected individuals will constitute a population called intermediate population. The crossing and mutation methods used were the BLX-α crossing and uniform mutation, with rates defined, respectively, at 85% and 1%, as suggested in literature [8].

6 Simulation Results

The number of stages and the number of states in each stage were increased step by step. These numbers were obtained by using the values of the integer set defined by {2, 4, 8, 16, 32, 64}. The values of the weights $d_{xi,(x+1)j}$, which link the i-th state of stage x to the j-th state of the following stage $(x + 1)$ were randomly selected of the integer set defined by {1, 3, 5, 7, 9}. For those instances with number of stages and states less than 32, each experiment has been simulated 20 times with random initial conditions and metric data. Other cases have been simulated ten times. Table 1 lists the simulation results. The columns n and m describe the number of stages and the number of state in each stage, respectively. The columns assigned as D^{NGA}, D^{MHA} and D^{DPN} show the average normalized path length obtained by the neurogenetic approach proposed

Table 1. Simulation results and comparative analysis

n	m	D^{NGA}	D^{MHA}	D^{DPN}	n	m	D^{NGA}	D^{MHA}	D^{DPN}
2	2	3.67	3.13	3.25	4	4	2.60	2.03	3.12
8	8	1.50	1.34	2.00	16	16	1.35	1.06	1.85
64	64	1.73	1.02	1.39	16	2	3.71	3.14	3.21
16	4	1.45	1.79	2.98	16	8	1.54	1.26	1.85
16	32	1.35	1.13	1.79	2	16	1.67	1.17	1.53
4	16	1.16	1.02	1.60	8	16	1.33	1.09	1.76

in previous sections and the results obtained by [4] and [9], respectively. The average normalized path length is defined by:

$$D = \frac{S_c}{n_s \cdot (n+1)} \tag{25}$$

where S_c is the sum of the selected paths, n_s is the number of simulations e n is the number of stages.

7 Conclusions

The main advantages of using the neurogenetic approach proposed in this paper are the following ones: i) the internal parameters of the network are explicitly obtained by the valid-subspace technique of solutions, ii) the valid-subspace technique groups all feasible solutions to the dynamic programming problem, iii) lack of need for adjustment of weighting constants for initialization, iv) for all classes of dynamic programming problems, a same methodology is adopted to derive the internal parameters of the network, and v) for industrial application, the neurogenetic approach offers simplicity of implementation in analog hardware.

References

1. Hopfield, J.J., Tank, D.W.: Neural Computation of Decisions in Optimization Problems. Biological Cybernetics 52, 141–152 (1985)
2. Zak, S.H., Upatising, V., Hui, S.: Solving Linear Programming Problems with Neural Networks. IEEE Trans. on Neural Networks. 6, 94–104 (1995)
3. Xia, Y., Feng, G.: A New Neural Network for Solving Nonlinear Projection Equations. Neural Networks 20, 577–589 (2007)
4. Silva, I.N., Goedtel, A., Flauzino, R.A.: The Modified Hopfield Architecture Applied in Dynamic Programming Problems and Bipartite Graph Optimization. International Journal of Hybrid Intelligent Systems 4, 17–26 (2007)
5. Hillier, F.S., Lieberman, G.J.: Introduction to Operations Research. Holden Day, San Francisco (1980)
6. Aiyer, S.V.B., Niranjan, M., Fallside, F.: A Theoretical Investigation into the Performance of the Hopfield Network. IEEE Trans. on Neural Networks 1, 204–215 (1990)
7. Graham, A.: Kronecher Products and Matrix Calculus. Ellis Horwood Ltd., Chichester (1981)
8. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Massachusetts (1996)
9. Chiu, C., Maa, C.Y., Shanblatt, M.A.: Energy Function Analysis of Dynamic Programming Neural Networks. IEEE Trans. on Neural Networks 2, 418–426 (1991)

Optimization of Parameters of Feed-Back Pulse Coupled Neural Network Applied to the Segmentation of Material Microstructure Images

Łukasz Rauch, Łukasz Sztangret, and Jan Kusiak

AGH – University of Science and Technology, al. Mickiewicza 30,
30-059 Krakow, Poland
{lrauch,szt,kusiak}@agh.edu.pl

Abstract. The paper presents application of bio-inspired optimization procedures to the problem of image segmentation of material microstructures. The method used for image processing was Feed-Back Pulse Coupled Neural Network (FBPCNN), which is very flexible in the case of highly diversified images, offering interesting results of segmentation. However, six input parameters of FBPCNN have to be adjusted dependently on image content to obtain optimal results. This was the main objective of the paper. Therefore, the procedure of image segmentation assessment was proposed on the basis of number of segments, their size, entropy and fractal dimension. The proposed evaluation was used as objective function in optimization algorithms. The results obtained for Simple Genetic Algorithms, Particle Swarm Optimization and Simulated Annealing are presented.

Keywords: FBPCNN, optimization, image processing, material microstructure.

1 Introduction

Analysis of metals behaviour in micro scale is important in the process of production technology design, especially in the case of highly advanced materials like modern steel grades e.g. TRIP, IF, DP. Such analysis requires innovative computational tools, which support numerical simulations of a single production processes as well as sophisticated production cycles. The recently developed approaches applied for these purposes are based on the concept of Digital Material Representation (DMR) [1], where modelling of microstructure phenomena plays crucial role. To obtain realistic results of simulations a material microstructure has to be represented explicitly with all its features like grains, subgrains, inclusions, etc. Generation of a microstructure representation, also called the ‘artificial microstructure generation’, is one of the most important algorithmic parts of the methodology based on the DMR. Various approaches are used for the purpose of artificial microstructure generation. The most commonly used method is Voronoi Tessellation, however other methods like Cellular Automata, Monte Carlo, Sphere Growth or Image Processing can also be found [2]. The proposed in the paper method of the artificial microstructure generation is based on the segmentation of a real microstructure photograph (image). Image segmentation

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 80–87, 2010.
© Springer-Verlag Berlin Heidelberg 2010

algorithms offer the highest reliability by mapping real material microstructure directly onto virtual geometry. Nevertheless, final quality of created structures depends on quality of input images, while most of them are noised, corrupted or blurred. These visible picture distortions affect performance of segmentation algorithms as well as further reliability of numerical simulations.

There are numerous image segmentation techniques (Section 2), however the FBPCNN [3] technique has been chosen in the present work. FBPCNN procedure is flexible in case of highly diversified images, which makes it competitive to other segmentation methods. The most demanding part of this approach is necessity of adjustment of the FBPCNN main parameters described in Section 3.2. These parameters have strong influence on the quality of the FBPCNN final results and simultaneously they depend nonlinearly on the analyzed image content. The proposed in the paper adjustment procedure of these parameters is based on the bio-inspired optimization methods, which cope well with the nonlinear problems and multimodal objective functions. Additionally, the new measure of segmentation quality is proposed.

The following Sections of the paper present the brief review of commonly used approaches to image segmentation, description of the FBPCNN architecture, applied optimization methods and obtained results.

2 Review of Image Segmentation Methods

The image segmentation algorithms are the most explored methods in the field of image processing. There are many publications in this research area. Existing approaches can be divided into several subgroups depending on the type of the algorithms e.g. template matching, edge detection, tracing as well as depending on the type of implemented numerical techniques e.g. matrices convolution, artificial intelligence or bio-inspired (neural networks, content-based processing, cognitive recognition), clustering, statistical approaches, nonlinear diffusion analysis. Such algorithms play important role not only in the analysis of the 2D pictures, but also in the signals, images and multidimensional data processing. However, their complexity grows respectively to increase of data dimensionality. In a case of the analysis of microstructure photographs, application of any of the mentioned above methods can be more or less successful. Thus, the review of the most recent papers regarding this field of research can be enumerated as follows:

- Convolution – in practice, well-known and widely applied methods based on derivative operators. Such methods are flexible and simple in implementation by using special kernel matrices e.g. Prewitt or Sobel to obtain effect of edge detection [4].
- Bio-inspired – one of the most popular methods is Watershed algorithm, which originates from natural solution of landscape and watersheds [5,6]. The idea of this method is based on the initial segmentation of data into disjoint areas, which in the next steps are filled successively with water puddles until two of them meet. The Authors proposed also the application of special type of Watershed method implemented by using CA [7], which offers high flexibility in case of various images, their shapes, colours, etc.

- Nonlinear diffusion – technique of image processing based on nonlinear diffusion and popularized by Perona and Malik [8]. This method was further modified and improved for applications in the area of texture-based segmentation. The example of such approach is presented in [9], where author proposed to measure the scale of texture by using nonlinear diffusion followed by the multi-channel statistical region active contour adaptation. The method can be seen as a kind of unsupervised segmentation, because parameters are not sensitive to different texture images.
- Clustering-based – these approaches are sufficient mainly for images without additional distortions, however their efficiency in case of even slightly noised data is very poor [10]. Thus, the modifications of such methods with other computational techniques like optimization procedures are often proposed [11]. The main advantage of this solution is insensibility for distortions, which offers much more reliable results.

The process of segmentation is very sophisticated and obtained results can be highly diversified even for the same input data and algorithms. This phenomenon depends mainly on the parameters established for the selected algorithm. Moreover, the automated assessment of the results is difficult, thus it is hard to design the unified segmentation method able to work on various types of images. It is often required in algorithms used during reconstruction of 3D artificial microstructure, which is based on the sequence of hundreds of microscopic pictures. The proposition of a method dedicated to assess segmentation results is presented in [12]. This approach consists of the framework based on Bayesian network, which determines optimal segmentation algorithm through a specific learning process. Another assessment method was proposed by Authors in [7], which is based on the calculation of fractal dimension offering slightly different quality measure of the results. In this paper the approach based on fractal dimension was enhanced with the entropies and sizes of image segments. The description of new measure is presented in Section 4.

3 Image Segmentation Using FBPCNN

3.1 FBPCNN Architecture

The FBPCNN networks are characterized by unique architecture. The neurons are organized in a cellular way as 2D matrix and each neuron is connected with set of neighbouring neurons. Such architecture originates directly from the PCNN networks proposed by Eckhorn, who observed oscillatory activities stimulated by external stimulus in cat's primary visual cortex [3]. The main result of this research was creation of neural model able to simulate mechanism of simplified visualization. Proposed model was recognized as having significant application potential in image processing and was finally adapted for the purpose of more sophisticated approaches. Each neuron, being the fundamental element of complex neural network, is made of three basic elements i.e. Linking Field, Feeding Field (Stimulus) and Pulse Generator (Figure 1).

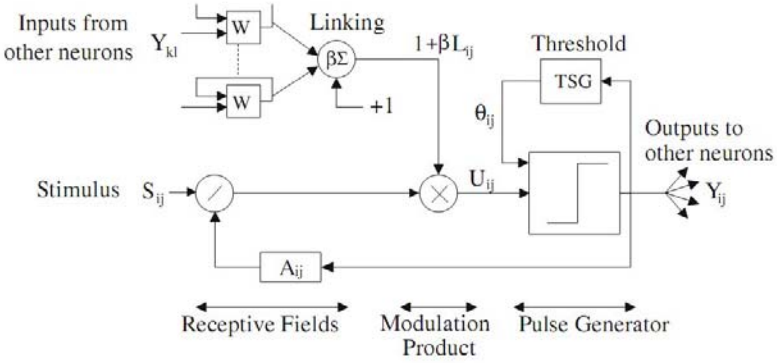


Fig. 1. The model of neuron applied in FBPCNN

The Linking and Feeding Fields are responsible for gathering signals generated by adjacent neurons. The role of Pulse Generator is similar to typical activation functionality, which generates Heaviside signal on the basis of input value and threshold. All variables presented in Figure 1 i.e. L_{ij}, , U_{ij}, Y_{ij}, θ_{ij}, S_{ij}, A_{ij} are defined by proper equations [3], which requires the following parameters: α_θ, V_θ, α_L, V_L, α_A, V_A, β. These parameters due to influence on FBPCNN performance and results are the main subject of the further optimization.

3.2 FBPCNN in Image Segmentation

FBPCNN architecture, because of its cellular character, can be easily mapped onto the structure of neighbouring pixels of an analyzed image. In that case, amount of neurons reflects the amount of pixels inside such image (Figure 2).

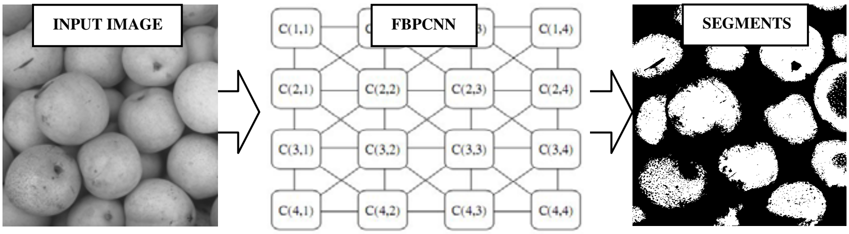


Fig. 2. Scheme of FBPCNN application in image segmentation

The network does not require learning process. It runs iteratively, activating or deactivating neurons in each iteration. Thus, the following results are presented in form of binary image. The basic challenge in application of the FBPCNN is the preprocessing stage, aiming at the establishment of proper values of mentioned earlier network parameters. The sensitivity analysis of such networks regarding the variation of weight parameters was presented by Authors in [13]. As the result of this work optimal solution for β parameter was determined for FBPCNN dedicated to processing of pictures of material microstructures. Those calculations offered the best results in case of β established dynamically for each neuron separately. The previous solution with β assumed to be constant was proved to be inefficient. Therefore, six remaining parameters still require to be adjusted i.e. α_θ , V_θ , α_L , V_L , α_A , V_A .

The adjustment of the network parameters can be solved by application of optimization procedures. On the purposes of this work, Authors selected bio-inspired methods. The input data is composed of the set of six parameters and original image, while the objective function is based on evaluation of results obtained from FBPCNN.

4 Optimization of FBPCNN Parameters

As mentioned in Section 2, the evaluation of segmentation is difficult, especially in case, when hundreds of results have to be assessed and the evaluation has to be done automatically (subjective assessment). The solution presented in this paper is based on calculations of entropy, fractal dimension as well as number and sizes of the segments. At first all the segments detected on the images are arranged into the list of N separated objects. Each of these objects is characterized by its entropy (E_j) and fractal dimension (FD_j). The entropy is calculated according to the following equation:

$$E_j = -\sum_i \frac{p_i}{N} \log \frac{p_i}{N} \tag{1}$$

where p_i is a number of pixels in cluster i related to specific colour and N is a number of pixels in segment j. FD_j is approximated by algorithm proposed by Gonzato [14]. The area of the segment is extended to square shape and iteratively covered by shorter side squares starting from large figures. The side of the square figure in i^{th} iteration (η_i) and the number of segments containing pixels ($N(\eta_i)$), form the following relation:

$$\log(N(\eta_i)) = -FD_j \log(\eta_i) + a \tag{2}$$

where FD_j is calculated as the slope of the regression line. The final evaluation of the segmentation process is composed of entropy (E_j), fractal dimension (FD_j), image size (I), size of segment (S_j) and weights (w_e, w_{fd}) and can be defined as follows:

$$Ev = \left(w_e \sum_j E_j + w_{fd} \sum_j FD_j \right) \frac{I}{\sum_j S_j} \tag{3}$$

The evaluation of segmentation quality is used further as the objective function in optimization algorithm. Three optimization techniques were selected to find the best values of six parameters of implemented FBPCNN model i.e. Simple Genetic Algorithm (SGA), Particle Swarm Optimization (PSO) and Simulated Annealing (SA). The bio-inspired optimization strategies were selected, because of multimodality and highly nonlinear character of the objective function as well as no information about its derivatives. The reliability of selected methods were proved in other material engineering applications [15]. During the optimization process the block limitations were imposed on all decision variables. The stop criteria for the number of objective function calls are defined by fixing the number of calls.

5 Results

The proposed methodology of combined FBPCNN and optimization methods was tested using sets of various microstructure images including dual phase (DP), TRIP

Table 1. The FBPCNN parameters determined in optimization process

Opt. method	α_θ	V_θ	α_L	V_L	α_A	V_A	*Ev*	*No. iter.*
DP								
SGA	5.443	0.539	-64.15	-3.443	7.197	0.647	**16.51**	**65**
PSO	8.614	7.192	95.40	4.693	7.733	7.218	22.03	1118
SA	0.000	4.018	169.6	-1.289	9.054	10.00	16.32	1514
TRIP								
SGA	0.410	2.743	2.731	7.180	5.369	-1.182	22.08	777
PSO	8.957	0.773	190.8	-3.883	5.703	6.021	**20.01**	**89**
SA	7.420	-3.374	-34.50	-8.073	6.746	6.911	19.25	1164
Bainitic								
SGA	6.318	-2.362	131.6	-4.975	8.487	9.517	**12.19**	**187**
PSO	4.345	5.855	163.0	0.488	5.601	0.785	30.04	247
SA	10.00	10.00	58.18	10.00	10.00	2.024	37.44	334

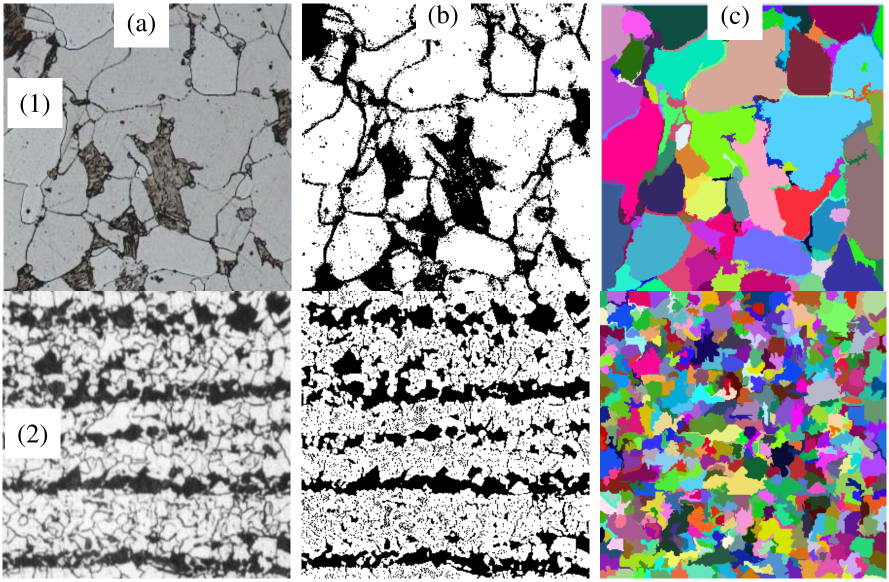


Fig. 3. Results obtained for microstructure of DP steels with large (1) and small (2) grains. Columns present original images (a), FBPCNN output (b) and final (after Watershed) segmentation results (c)

and bainitic steels. For each set of microscopic pictures the optimization calculations were performed using each of selected methods. The FBPCNN parameters determined for each set are presented in table 1.

In most cases the tested optimization procedures were able to find optimal solutions. The best results were determined by SGA and PSO, while SA always required much more iteration and time to find results with comparable quality. The results obtained from optimized FBPCNN were passed as input data to Watershed algorithm [7] to perform final stage of segmentation. FBPCNN results were used as starting points for flooding procedure. The examples of FBPCNN and Watershed results are presented in Figure 3.

6 Conclusions

The paper presents approach to image segmentation based on FBPCNN combined with optimization algorithms. Three different optimization procedures were used to determine the best values of the FBPCNN parameters i.e. SGA, PSO and SA. The parameters were established for different sets of microscopic pictures of different steel grades. The segmentation quality (eq. 3) was proposed as the optimization objective function. This measure was created on the basis of segments' entropy, fractal dimension and average size.

The obtained results proved satisfactory reliability of FBPCNN as the first stage of segmentation. Pixels activated by FBPCNN were passed as input signals to the Watershed algorithm and used for final segmentation. Obtained segments can be further applied in numerical simulation of digital material representation, which offers interesting information about material behaviour in micro scale. Thus, the future objectives of the work will be focused on detailed analysis of segmented images of other materials e.g. determination of micro inclusions, scratches, etc.

Acknowledgments. The financial support of Polish Ministry of Science and Higher Education, AGH project no. 11.11.110.861 is acknowledged.

References

1. Bernacki, M., Chastel, Y., Digonnet, H., Resk, H., Coupez, T., Logé, R.E.: Development of numerical tools for the multiscale modeling of recrystallisation in metals, based on a digital material framework. Computer Methods in Materials Science 7(1), 142–149 (2007)
2. Cybulka, G., Jamrozik, P., Wejrzanowski, T., Rauch, L., Madej, L.: Digital representation of microstructure. In: Proc. CMS Conf., Krakow, pp. 379–384 (2007)
3. Eckhorn, R.: Oscillatory and non-oscillatory synchronizations in the visual cortex and their possible roles in associations of visual features. Progress in Brain Research 102, 405–426 (1994)
4. Nixon, M.S., Aguado, A.S.: Feature extraction and image processing, 1st edn., Newnes (2002)
5. Bleau, A., Leon, L.J.: Watershed-Based segmentation and region merging. Computer Vision and Image Understanding 77, 317–370 (2000)

6. Zhao, C.G., Zhuang, T.G.: A hybrid boundary detection algorithm based on watershed and snake. Pattern Recognition Letters 26, 1256–1265 (2005)
7. Rauch, L., Straus, M.: Implementation of watershed algorithm based on cellular automata combined with estimation of 2D fractal dimension. In: Proc. of SEECCM 2009 European Conf. on Computational Mechanics, pp. 89–96 (2009)
8. Perona, P., Malik, J.: Scale space and edge detection using anisoropic diffusion. J. IEEE Transactions on Pattern Analysis and Machine Intelligence 12, 629–639 (1990)
9. Zhang, Y.: Texture image segmentation based on nonlinear diffusion. Geo-spatial Information Science 11(1), 38–42 (2008)
10. Liew, A.W., Yan, H., Law, N.F.: Image segmentation based on adaptive cluster prototype estimation. J. IEEE Transactions on Fuzzy Systems 13(4), 444–449 (2005)
11. Zhou, X.C., Shen, Q.T., Liu, L.M.: New two-dimensional fuzzy C-means clustering algorithm for image segmentation. Journal of Central South University of Technology 15, 882–887 (2008)
12. Shah, S.K.: Performance modeling and algorithm characterization for robust image segmentation. International Journal of Computer Vision 80, 92–103 (2008)
13. Lukasik, L., Rauch, L.: Estimation of parameters of Feed-Back Pulse Coupled Neural Networks (FBPCNN) for purposes of microstructure images segmentation. Computer Methods in Materials Science (2009) (in print)
14. Gonzato, G.: A practical implementation of the box counting algorithm. Computers and Geosciences 24(1), 95–100 (1998)
15. Sztangret, L., Stanislawczyk, A., Kusiak, J.: Bio-inspired optimization strategies in control of copper flash smelting process. Computer Methods in Materials Science 9(3), 400–408 (2009)

Hybrid Neural Networks as Prediction Models

Izabela Rojek

Kazimierz Wielki University in Bydgoszcz,
Institute of Mechanics and Applied Computer Science,
Chodkiewicza 30, 85-064 Bydgoszcz, Poland
izarojek@ukw.edu.pl
http://www.imsis.ukw.edu.pl

Abstract. The paper presents hybrid neural networks as prediction models for water intake in water supply system. Previous research concerned establishing prediction models in the form of single neural networks: linear network (L), multi-layer network with error back propagation (MLP) and Radial Basis Function network (RBF). Currently, the models in the form of hybrid neural networks (L-MLP, L-RBF, MLP-RBF and L-MLP-RBF) were created. The prediction models were compared for obtaining optimal prognosis. Prediction models were done for working days, Saturdays and Sundays. The research was done for selected nodes of water supply system: detached house node and nodes for 4 hydrophore stations from different pressure areas of water supply system. Models for Sundays were presented in detail.

Keywords: prediction model, hybrid neural network, water supply system.

1 Introduction

Current monitoring systems of water supply system lack the models for forecasting network load. The network load changes in cycles, depending on the season of the year, day of the week or time of day. These influence the way the water supply system is managed, especially as regards the filling or emptying of impounding tanks located in the network. When predicting greater network load, the tanks have to be filled accordingly, or could be emptied in the case of a lesser need for water.

Water network is central to the municipal system of water supply and sewage. The load of water network influences the work of pumps in the water intake and treatment station, the load of sewage system and the load of sewage treatment plant [1].

Accurate forecasting of load and operation of the water supply system will allow energy efficient operation of pumps at the water intake, as well as effective management of the technological process in the sewage treatment plant by preparing it for a certain load of sewage and pollution. That is why the most important issue is the long term forecast of the load of water supply system.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 88–95, 2010.
© Springer-Verlag Berlin Heidelberg 2010

In the contemporary world, people, as well as enterprises are flooded by data or information from many sources. Unfortunately this valuable data, the accumulation and processing of which costs millions, is usually lost in databases and data warehouses. The problem stems form the lack of qualified analysts who could convert data into knowledge. Detailed analysis of information might prove very useful in establishing a relation with a client, and adapting an offer for the client's special needs. In this case, traditional forecasting methods are insufficient. That is why data mining is essential nowadays [2].

Earlier research done by the author, concerned construction of prediction models in the form of single neural networks: linear network (L), multi-layer network with error backpropagation (MLP) and Radial Basis Function network (RBF).

In papers we meet the different applications of hybrid neural networks [3], [4], [5], [6]. They give better solutions than simple neural networks. Therefore the author created forecasting models in the form of hybrid neural networks .

In the paper, different types of hybrid neural networks were applied to forecast the load of water network. The experiments had two aims. Firstly, the paper presents and compares forecasting models in the form of selected types of hybrid neural network: L-MLP, L-RBF, MLP-RBF and L-MLP-RBF. After comparison, we can say which type of neural network is the best for prediction of load of water network. Secondly, the research was done for selected nodes of water supply system: a detached house node and nodes for 4 hydrophore stations from different pressure areas of water supply system. Specific nodes were selected for checking whether different networks nodes require different prognosis models, or whether one shall suffice.

2 Hybrid Neural Network as a Forecasting Model

Neural networks return continuous value at the output, hence they are excellent for estimating and forecasting. Such networks may analyze many variables at a time. It is possible to create a model even if the solution is very complex. The drawbacks of neural networks are the difficulties in setting architectural parameters, falling into local minima, long learning process and the lack of clear interpretation [7].

Hybrid models constructed of different neural networks were drawn up (linear neural network, multi-layer neural network network with error backpropagation and Radial Basis Function network). The output for the hybrid model is composed of different network outputs. Two types of neural network sets were considered. In the first case, for classification purposes, the first prediction is obtained through voting (the winner takes it all) - the most represented value is the starting value for the set (a set with a winner). In the second case, the component networks are limited to a certain constraint. The complex output is formed on the level of output neurons. The sets of this kind average the outputs in all component networks (a set with an average).

Sets equipped with an important tool against the overlearning of the network; these improve the generalisation possibilities for the model. Averaging of predictions obtained from the networks of different structure, taught in a different way - on the basis of other cases, decreases the range of results. That is a simple way of improving the generalization skills. Theory shows that the quality of the set is better or at least equal to the quality of component networks. The quality of hybrid neural networks was assessed by cross-validation. These models were compared with a view to obtaining an optimal model to control the parameters of the water networks. The prediction models were drawn up for predicting the load of the water network in different time intervals: hour, day, week, month and season. Each model was tested for the correct prediction.

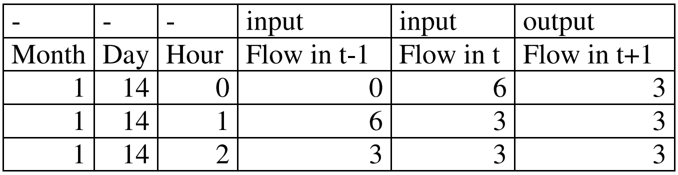
The models were taught correct prediction on the basis of the real data obtained from the water supply company from Rzeszow. The prediction models and the generated decision rules allowed to predict the changes in the load of a water network. These models allow for current corrections to the work of pumps, in order to obtain the predicted demand for water. The correctness of prediction models was assessed. The first phase of assessing models was completed during construction of models in the form of neural networks, where RMS error was checked as well as the coefficient of number of patterns above tolerance. The results obtained from prediction models were also compared with the real values of water network load.

3 Hybrid Neural Model for Prediction of Water Intake in a Water Supply System Node

Experiments were conducted for selected water supply system nodes: detached house node, nodes for 4 hydrophore stations from different pressure areas of water supply system. For each node, three models were drawn: working day, Saturday, Sunday, using hybrid neural networks (L-MLP, L-RBF, MLP-RBF, L-MLP-RBF). For each type of network, a research examining which network produced best results was done. Analysis and comparison of prediction models for water intake was carried out. The paper was illustrated with graphs presenting data from a Sunday.

3.1 Data Preparation

The measured data presents the state of water supply system in the selected measuring nodes. *Presentation of data* - measurement was taken every hour. The data is transferred to database by PlusGSM mobile communications. Data was processed through clearing and conversion. Data was divided into periods: working day, Saturday and Sunday. The missing values were put. The data is used as learning files for prediction models. *Selection of learning data* - data is grouped according to month, day and hour. Separate learning files were created for every node. Data includes real measurements of water flow from Rzeszow water supply and sewage company. The file contained data for the water flow during 3 months. Knowing the model of the facility, the reaction to various input violations should be analysed. Data were divided into the learning set (80% of data) and the testing set (20% of data). Figure 1 gives a fragment of a learning file. It is interesting to define the future state of the facility for the time *t+n*, where *n* is the prediction horizon, *t* contains the input changes history up to the present. In order to construct time sequences which are later used in the forecasting model, the values of flow before the moment *t* (Flow t-1) and after the moment *t* (Flow t+1) were added. Neural network *inputs* are the following: flow t-1 and flow t with take into account month, day and hour. *Output* is value of water flow t+1.



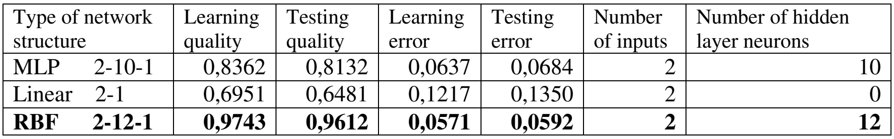
-	-	-	input	input	output
Month	Day	Hour	Flow in t-1	Flow in t	Flow in t+1
1	14	0	0	6	3
1	14	1	6	3	3
1	14	2	3	3	3

Fig. 1. Learning file fragment

3.2 Drawing a Hybrid Neural Network Model for Forecasting

The stages of prognosis procedure are as follows: observation of time sequence process, creating a model of the observed sequence and conversion of the time sequence model into a shape which enables prognosis (construction of forecasting algorithm).

The paper [8] presents earlier research concerning single models (L, MLP and RBF). These models were tested on selected network nodes, and the Radial Basic Function model proved to be the most accurate. Figure 2 shows that parameters of RBF network are smallest.The next phase of research concerned drawing up of hybrid neural network models.



Type of network structure	Learning quality	Testing quality	Learning error	Testing error	Number of inputs	Number of hidden layer neurons
MLP 2-10-1	0,8362	0,8132	0,0637	0,0684	2	10
Linear 2-1	0,6951	0,6481	0,1217	0,1350	2	0
RBF 2-12-1	**0,9743**	**0,9612**	**0,0571**	**0,0592**	**2**	**12**

Fig. 2. Parameters describing experiments for the best MLP, Linear and RBF networks

The hybrid neural network models were drawn. The models are equipped with two inputs *(Flow t-1, Flow)* and one output *(Flow t+1)*.

These structures were taught with different conditions of ending the process, i.e. the end after reaching the number of periods equal to 1000, 10000 or 100000. To every combination, a RMS error was compared (Formula 1).

$$RMS = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(T_i - O_i)^2} \quad (1)$$

where: n - number of instances, T_i - pattern value, O_i - real value.

Hybrid model of the linear (L) and MLP network. The following experiments as regards the hybrid model constructed of a linear (L) and MLP network. For the network set, the following parameters for description of the experiment were obtained: learning quality (0,7657), testing quality (0,7307), learning error (0,0927), testing error (0,1017).

Figure 3 illustrates the forecast using the hybrid network (L-MLP).

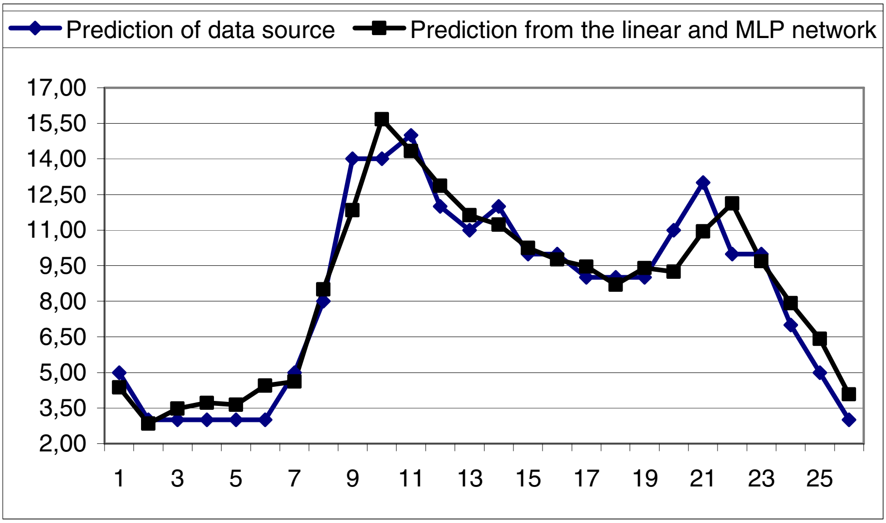


Fig. 3. The forecast using the hybrid neural network (L-MLP) for the node of Budziwoj - Rzeszow (X axis presents weeks, Y axis - values of water flow)

Hybrid model of the linear (L) and RBF network. The following experiments as regards the hybrid model constructed of a linear (L) and RBF network. For the network set, the following parameters for description of the experiment were obtained: learning quality (0,8347), testing quality (0,8047), learning error (0,0894), testing error (0,0971).

Hybrid model of the MLP and RBF network. The following experiments as regards the hybrid model constructed of a MLP and RBF network. For the network set, the following parameters for description of the experiment were obtained: learning quality (0,9053), testing quality (0,8872), learning error (0,0604), testing error (0,0638).

Figure 4 illustrates the forecast using the hybrid network (MLP-RBF).

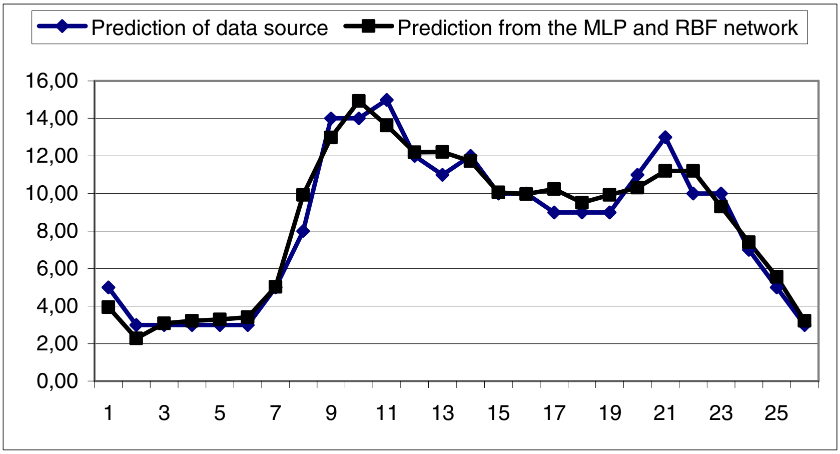


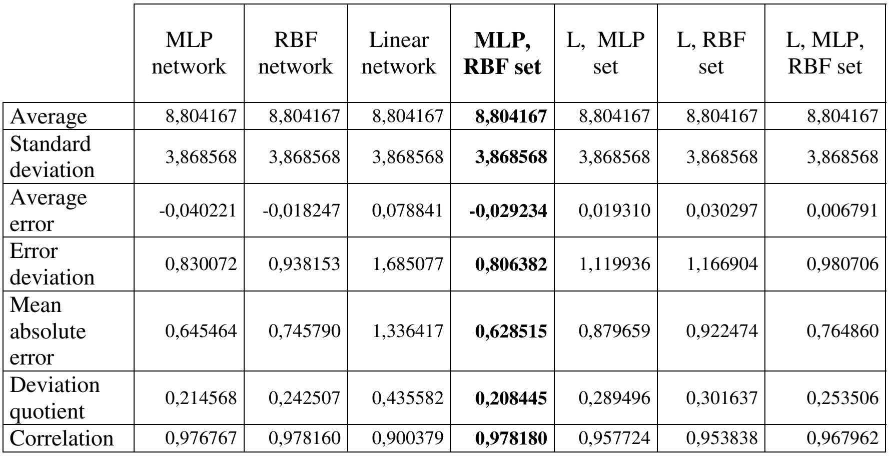
Fig. 4. The forecast using the hybrid neural network (MLP-RBF) for the node of Budziwoj - Rzeszow (X axis presents weeks, Y axis - values of water flow)

Hybrid model of the linear (L), MLP and RBF network. The following experiments as regards the hybrid model constructed of a linear (L), MLP and RBF network. For the network set, the following parameters for description of the experiment were obtained: learning quality (0,8352), testing quality (0,8075), learning error (0,0803), testing error (0,0875).

Assessment of the prediction models. The assessment of the quality of neural prediction models may be done by comparing the graphs: the real and the forecast. It is a very common form of presenting research results. Yet the quantitative methods for assessment of neural models are the ones which allow to formulate more objective conclusions. The assessment of neural models is generally carried out in two phases. In the first phase, after having constructed the taught networks, assessment is done through the so-called regressive statistics. Figure 5 shows regressive statistics for weekly prediction models.

The quotient of standard deviation and correlation of real and forecasted values are the most important for assessment of neural models [9]. The first parameter for the constructed models should assume the values of 0,1-0,2.
The deviation quotient with a value close to zero testifies to a good value of a given model. If this is greater than one (or close to one), then the designed model may be rejected. It is difficult to clearly define the correctness of the model if the deviation quotient falls within the range of 0,3-0,7. The quality of the model must eventually be obtained form the obtained ex post errors, or their acceptability in the particular case. The correlation between the real and forecasted values assumes values form 0-1. It is best when close to one (the closer, the better).

The quotient of standard deviation = quotient error deviation / standard deviation of real data.



	MLP network	RBF network	Linear network	**MLP, RBF set**	L, MLP set	L, RBF set	L, MLP, RBF set
Average	8,804167	8,804167	8,804167	**8,804167**	8,804167	8,804167	8,804167
Standard deviation	3,868568	3,868568	3,868568	**3,868568**	3,868568	3,868568	3,868568
Average error	-0,040221	-0,018247	0,078841	**-0,029234**	0,019310	0,030297	0,006791
Error deviation	0,830072	0,938153	1,685077	**0,806382**	1,119936	1,166904	0,980706
Mean absolute error	0,645464	0,745790	1,336417	**0,628515**	0,879659	0,922474	0,764860
Deviation quotient	0,214568	0,242507	0,435582	**0,208445**	0,289496	0,301637	0,253506
Correlation	0,976767	0,978160	0,900379	**0,978180**	0,957724	0,953838	0,967962

Fig. 5. Regressive statistics for weekly prediction models

Correlation is practically standard Pearson correlation coefficient r (Formula 2).

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}} \qquad (2)$$

where: $\overline{x}, \overline{y}$ - average values, x_i, y_i - values of random tests.

In the second phase, after having taught the network, the forecasting process is carried out. Except the real values, the prediction of output variable for the model is obtained. It allows to determine the ex post errors. The above defined quality measurements of the neural model prediction errors were given in the experiment phase. The value of the RMS errors of learning and testing were given with the drawn up models. Neural networks generalized well, because answers given by them were comprised in the range of error for testing file.

4 Conclusion

The general principle of science states that if there is a possibility to choose between a complex and a basic model, the basic model should always be given priority - unless of course the more complex one is significantly better for the particular data than the basic one. This principle should also be applied to neural networks. Having analyzed various forecasting models, it was the MLP-RBF network model which proved the most effective. Since these networks are similar in adjustment when it comes to choosing a network to create the forecasting model, the complexity of a model should also be taken into account - the learning time, calculation time and accuracy. We have selected the MLP-RBF network model for further research. Further research concerning the creation of forecasting models should be directed towards constructing models not only for weeks, but also for the four seasons of the year: spring, summer, autumn and winter, and finally the entire year. Having constructed all these models, a full model for forecasting the load of water network at any time shall be obtained. When acquiring new models, these models should be taught in any moment in time. Further research into when and how to teach these models should be done.

References

1. Studzinski, J., Bogdan, L.: Computer Aided Decisions Making System for Management, Control and Planning Water and Wastewater Systems. In: Applications of Informatics in Science, Engineering and Management, System Research Institute, Polish Academy of Sciences, Warsaw. Systems Research, vol. 49, pp. 149–157 (2006)
2. Michalski, R.S., Bratko, I., Kubat, M.: Machine Learning and Data Mining. John Wiley & Sons, Chichester (1998)
3. Smaoui, N.: A Hybrid Neural Network Model for the Dynamics of the Kuramoto-Sivashinsky Equation. In: Mathematical Problems in Engineering, vol. 3, pp. 305–321. Hindawi Publishing Corporation (2004)
4. Caciotta, M., Giarnetti, S., Leccese, F.: Hybrid Neural Network System for Electric Load Forecasting of Telecomunication Station. In: XIX IMEKO World Congress Fundamental and Applied Metrology, Lisbon, pp. 657–661 (2009)
5. Tsai, C.-F., McGarry, K., Tait, J.: Image Classification Using Hybrid Neural Networks. In: ACM Conference on Research and Development in Information Retrieval, New York, pp. 431–432 (2003)
6. Chen, H., Grant-Muller, S., Mussone, L., Montgomery, F.: A Study of Hybrid Neural Network Approaches and the Effects of Missing Data on Traffic Forecasting. Journal Neural Computing and Applications 10(3), 277–286 (2001)
7. Krawiec, K., Stefanowski, J.: Machine Learning and Neural Networks. Publishing House of Poznan University of Technology, Poznan (2004) (in Polish)
8. Rojek, I.: Neural Networks as Prediction Models for Water Intake in Water Supply System. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 1109–1119. Springer, Heidelberg (2008)
9. Tadeusiewicz, R., Lula, P.: Statistica Neural Networks 4.0 PL: Introduction to neural networks. StatSoft Polska, Cracow (2001) (in Polish)

Fast Robust Learning Algorithm Dedicated to LMLS Criterion

Andrzej Rusiecki

Wroclaw University of Technology, Wroclaw, Poland
andrzej.rusiecki@pwr.wroc.pl

Abstract. Robust neural network learning algorithms are often applied to deal with the problem of gross errors and outliers. Unfortunately, such methods suffer from high computational complexity, which makes them ineffective. In this paper, we propose a new robust learning algorithm based on the LMLS (Least Mean Log Squares) error criterion. It can be considered, as a good trade-off between robustness to outliers and learning efficiency. As it was experimentally demonstrated, the novel method is not only faster but also more robust than the LMLS algorithm. Results of implementation and simulation of nets trained with the new algorithm, the traditional backpropagation (BP) algorithm and robust LMLS method are presented and compared.

1 Introduction

Feedforward neural networks (FFNs), considered as universal and model-free approximators [7], very often find their application in areas such as pattern recognition, function approximation, or signal and image processing. They are valued for the simple use, because they do not require prior analytical knowledge about the modelled system. FFNs can be trained by minimizing an error function on the training set, to fit the data as close as possible. Many authors noted however, that the performance of such learning methods depends strongly on the quality of the training data [5,8,11]. If the data set is corrupted by the large noise, or if outliers and gross errors appear, the network builds a model that can be very distant form the desired one. It happens because the backpropagation learning algorithm (and many of its variations) uses an error function based on the least mean squares method, which is optimal only for the normal error distribution.

Unfortunately, in many real-world cases, the assumption that the errors are normal and *iid* doesn't hold. Because of measurement errors, long-tailed noise, or human mistakes, data obtained from the environment are often affected by the large noise of unknown form or different types of gross errors. In fact, the quantity of such outliers ranges from 1 to even 10% [5]. Outliers, in other words, observations significantly deviating form the bulk of data, lie in the field of interest of so-called robust statistics [5,8]. Robust statistical methods and robust estimators are designed to act well when true underlying model deviates from the assumed parametric model.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 96–103, 2010.
© Springer-Verlag Berlin Heidelberg 2010

To deal with the problem of outliers several robust neural network learning algorithms have been proposed. Many of them make use of the idea of robust estimators [8], replacing quadratic function with a new loss error function, which helps in reducing the impact of outliers on the network training process without changing the idea of the backpropagation method.

Probably the first and the most often cited robust learning algorithm is the LMLS (Least Mean Log Squares) robust algorithm introduced by Liano [9]. He proposed the logistic error function, derived from the assumption of the errors generated with the Cauchy distribution. Other authors tried to use more sophisticated functions. The Hampel's hyperbolic tangent as a new error criterion was used by Chen and Jain [1]. The error function contains here a special parameter - scale estimator β, that defines the interval supposed to contain only clean data, depending on the assumed quantity of outliers or current errors values. Chunag and Su [2] combined the idea with the annealing concept applying the annealing scheme to decrease the value of β, whereas Pernia-Espinoza et al. [12] presented an error function based on robust tau-estimates. An approach based on the MCD estimator was also proposed [13].

Robust learning algorithms may often improve the network performance for corrupted training sets. However, they cannot be considered as universal tools, because of the properties of the robust estimators, on which they are based. Moreover, the error functions are often complex and difficult to compute [12,13] this is why there is still a need for new solutions. The performance of new robust learning methods is usually compared with the LMLS algorithm [1,2,12] because of its simplicity and relatively good efficiency. Moreover, for this algorithm it is not necessary (unlike some of the others) to use a few epochs of traditional method at the beginning of the training process. For that reason the LMLS method can be considered, as referential for robust learning algorithms. But also for this method some improvements can be made. In this article, we propose a new Levenberg-Marquardt-inspired robust learning algorithm, dedicated for the LMLS error criterion. The novel method is faster and more effective than the LMLS and simultaneously it is not as sophisticated as other existing algorithms.

2 Fast LMLS-Based Learning Algorithm

2.1 Robust LMLS Learning Algorithm

Liano [9] proposed a new loss function, called Least Mean Log Squares (LMLS), to introduce a robust error measure. The LMLS function was given as:

$$\rho(r_i) = \log(1 + \frac{1}{2}{r_i}^2), \tag{1}$$

where r_i is an error for the i-th training pattern. For the loss function, the influence function, that measures how data errors can affect the training process, is bounded and can be written as:

$$\psi(r_i) = \frac{r_i}{1 + \frac{1}{2}{r_i}^2}. \tag{2}$$

When the residuals are small, the function is close to linear but when they increase, the function decreases asymptotically towards zero:

$$\lim_{r->\infty} \frac{r_i}{1+\frac{1}{2}{r_i}^2} = 0, \tag{3}$$

so the impact of the largest errors on the criterion function value is limited.

The LMLS algorithm was originally proposed for the on-line learning type, where network weights are updated after presentation of each training pattern. It can be easily generalized to the batch learning and in such case the whole network error in a certain epoch can be written as:

$$E(\boldsymbol{w}) = \sum_{k=1}^{N}\sum_{i=1}^{m} \log(1+\frac{1}{2}{r_{ki}}^2(\boldsymbol{w})), \tag{4}$$

where $r_{ki} = (y_{ki}(\boldsymbol{w}) - t_{ki})$ is the error of i-th output for the k-th training set element and m is the number of network outputs. For the error criterion function defined by (4) one of the gradient learning algorithms can be used. Similarly to other robust learning methods we can apply a simple (and slow) gradient-descent learning algorithm or more sophisticated conjugate-gradient algorithm. Nevertheless it is impossible to use here, one of the most effective, Levenberg-Marquardt algorithm [10,4] because it is dedicated to quadratic error function.

2.2 Derivation of the Fast LMLS-Based Learning Algorithm

The algorithm is similar to the Levenberg-Marquardt in the process of computing single learning step, based on the Jacobian matrix. However, the rest of the method is significantly different and can be written as follows. To design a fast optimization algorithm for the error function given by (4) we can write the Jacobian matrix of output errors J:

$$J(\boldsymbol{w}) = \begin{bmatrix} \frac{\partial r_{11}}{\partial w_1} & \frac{\partial r_{11}}{\partial w_2} & \cdots & \frac{\partial r_{11}}{\partial w_n} \\ \frac{\partial r_{21}}{\partial w_1} & \frac{\partial r_{21}}{\partial w_2} & \cdots & \frac{\partial r_{21}}{\partial w_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial r_{N1}}{\partial w_1} & \frac{\partial r_{N1}}{\partial w_2} & \cdots & \frac{\partial r_{N1}}{\partial w_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial r_{1m}}{\partial w_1} & \frac{\partial r_{1m}}{\partial w_2} & \cdots & \frac{\partial r_{1m}}{\partial w_n} \\ \frac{\partial r_{2m}}{\partial w_1} & \frac{\partial r_{2m}}{\partial w_2} & \cdots & \frac{\partial r_{2m}}{\partial w_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial r_{Nm}}{\partial w_1} & \frac{\partial r_{Nm}}{\partial w_2} & \cdots & \frac{\partial r_{Nm}}{\partial w_n} \end{bmatrix}, \tag{5}$$

where

$$r(\boldsymbol{w}) = \left[r_{11} \ldots r_{N1} \ldots r_{1m} \ldots r_{Nm} \right]^T, \tag{6}$$

and n is the number of network weights. The Jacobian matrix is also employed by the original Levenberg-Marquardt method [4] because its computation involves only slight modification to the backpropagation algorithm.

Now, we introduce matrix p, defined as:

$$p(\boldsymbol{w}) = \left[\tfrac{r_{11}}{1+\frac{1}{2}{r_{11}}^2} \; \tfrac{r_{21}}{1+\frac{1}{2}{r_{21}}^2} \cdots \tfrac{r_{Nm}}{1+\frac{1}{2}{r_{Nm}}^2} \right]^T . \tag{7}$$

As one can see, the matrix p is based on the vector of residuals after some simple algebraic operations. For the Jacobian matrix defined by (5), we can write the gradient of error function as:

$$\nabla E(\boldsymbol{w}) = J(\boldsymbol{w})^T p(\boldsymbol{w}). \tag{8}$$

Then we define a new matrix based on the elements of the vector r:

$$\begin{aligned} q(\boldsymbol{w}) = [& \tfrac{1}{1+\frac{1}{2}{r_{11}}^2} - \tfrac{{r_{11}}^2}{(1+\frac{1}{2}{r_{11}}^2)^2} \; \tfrac{1}{1+\frac{1}{2}{r_{21}}^2} - \tfrac{{r_{21}}^2}{(1+\frac{1}{2}{r_{21}}^2)^2} \\ & \cdots \tfrac{1}{1+\frac{1}{2}{r_{Nm}}^2} - \tfrac{{r_{Nm}}^2}{(1+\frac{1}{2}{r_{Nm}}^2)^2}]^T . \end{aligned} \tag{9}$$

We introduce matrix JQ by multiplying each row of the Jacobian matrix by the corresponding element of $q(\boldsymbol{w})$:

$$JQ(\boldsymbol{w}) = \begin{bmatrix} \frac{\partial r_{11}}{\partial w_1} q_{11} & \frac{\partial r_{11}}{\partial w_2} q_{11} & \cdots & \frac{\partial r_{11}}{\partial w_n} q_{11} \\ \frac{\partial r_{21}}{\partial w_1} q_{22} & \frac{\partial r_{21}}{\partial w_2} q_{22} & \cdots & \frac{\partial r_{21}}{\partial w_n} q_{22} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial r_{N1}}{\partial w_1} q_{N1} & \frac{\partial r_{N1}}{\partial w_2} q_{N1} & \cdots & \frac{\partial r_{N1}}{\partial w_n} q_{N1} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial r_{1m}}{\partial w_1} q_{1m} & \frac{\partial r_{1m}}{\partial w_2} q_{1m} & \cdots & \frac{\partial r_{1m}}{\partial w_n} q_{1m} \\ \frac{\partial r_{2m}}{\partial w_1} q_{2m} & \frac{\partial r_{2m}}{\partial w_2} q_{2m} & \cdots & \frac{\partial r_{2m}}{\partial w_n} q_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial r_{Nm}}{\partial w_1} q_{Nm} & \frac{\partial r_{Nm}}{\partial w_2} q_{Nm} & \cdots & \frac{\partial r_{Nm}}{\partial w_n} q_{Nm} \end{bmatrix} . \tag{10}$$

Now, we are able to write the Hessian matrix:

$$H(\boldsymbol{w}) = J(\boldsymbol{w})^T JQ(\boldsymbol{w}) + S(\boldsymbol{w}). \tag{11}$$

The estimation of the Hessian is therefore:

$$H(\boldsymbol{w}) \approx J(\boldsymbol{w})^T JQ(\boldsymbol{w}). \tag{12}$$

The skipped element S depends on the second derivative and can be written as:

$$S(\boldsymbol{w}) = \sum_{k=1}^{N} \sum_{i=1}^{m} p_{ki}(\boldsymbol{w}) \nabla^2 r_{ki}(\boldsymbol{w}) \tag{13}$$

Similarly to the Leveberg-Marquardt algorithm we introduce a regularisation factor u and finally we can write the Hessian matrix as:

$$H(\boldsymbol{w}) = J(\boldsymbol{w})^T JQ(\boldsymbol{w}) + u(t)1. \tag{14}$$

In each training epoch t, parameter $u(t)$ is multiplied by η whenever the error would increase until it finally decreases. Then $u(t)$ is divided by β for the next epoch. Following [4] we set $u(0) = 0.001$, $\eta = 10$, and $\beta = 10$.

The network weights in every training epoch are then updated as follows:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \left[J(\boldsymbol{w}_t)^T JQ(\boldsymbol{w}_t) + u(t)1\right]^{-1} \nabla E(\boldsymbol{w}_t). \tag{15}$$

3 Simulation Results

Robust learning algorithms are particularly useful in function approximation. To test the performance of the proposed learning algorithm we prepared many examination tasks. Results obtained for two of them are presented below. The first function to be approximated is $y = x^{-2/3}$ proposed in [1]. Many authors tested their algorithms on this function [9,2,12] because of its shape in the neighbourhood of a point (0,0). The second one is a two-dimensional spiral given as $x = \sin y$, $z = \cos y$.

To simulate large noise and outliers present in real data we used two models:

- Gross Error Model (Type 1): Clean data are corrupted with additive noise $F = (1-\delta)G + \delta H$, where F is the error distribution, $G \sim N(0.0, 0.1)$ and $H \sim N(0.0, 10.0)$ are Gaussin noise occurring with probability $1-\delta$ and outliers with probability δ.
- High Value Random Outliers [12] (Type 2): Data are corrupted with outliers of the form $F = (1-\delta)G + \delta(H_1 + H_2 + H_3 + H_4)$, where: $H_1 \sim N(15, 2)$, $H_2 \sim N(-20, 3)$, $H_3 \sim N(30, 1.5)$, $H_4 \sim N(-12, 4)$.

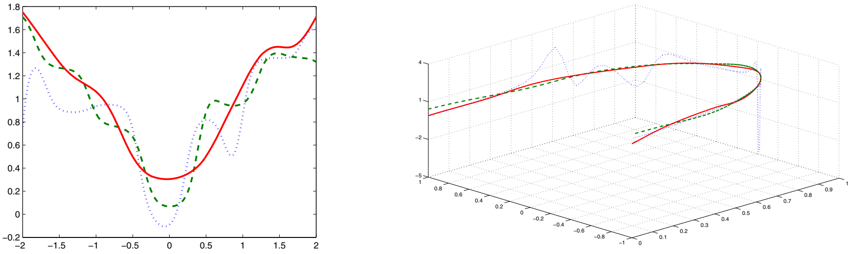


Fig. 1. Simulation results for the network trained with different algorithms (Data Type 1, $\delta = 0.2$): backpropagation algorithm (*dotted line*), LMLS algorithm (*dashed line*), FLMLS algorithm (*solid line*)

3.1 Results

We have compared the performance of traditional backpropagation algorithm (BP), robust LMLS algorithm and our new fast LMLS-based robust learning algorithm (FLMLS). Discussions and comparisons of the LMLS criterion with other, more complicated robust learning algorithms may be found in [2,12,13]. The tested algorithms were applied to the simple three-layer feedforward neural network, having one or two inputs (for one- or two-dimensional problem), ten neurons in the hidden layer and one output. To make the comparison reliable the batch-type learning for the LMLS algorithm was used. As the optimization method we've employed here the conjugate gradient method [3]. Testing data were generated simply as points lying on the approximated curves. The network performance was tested 500 times for each *problem - probability of outliers* combination. The simulation results: the average MSE and standard deviation over 500 randomly generated initial weights sets and outlying points sets were gathered in tables.

Looking at the Table 1 and 2 one may notice, that whereas for the clean data all tested algorithms act relatively well (though the FLMLS causes the highest error level), when the training data contained noise and outliers, as it was expected, the backpropagation algorithm is definitely worse than two variations of the method with LMLS error function (an example of the network response can be found in the Fig. 1). What is interesting, is that the errors of the networks trained with our novel FLMLS method are in each case slightly lower than for the classic LMLS algorithm. The situation is similar for the data contaminated with high value outliers (Table 2). Here, the performance of the BP algorithm is intolerable. After analysing results obtained for the two-dimensional approximation (Table 3), we can see that the difference between the LMLS and FLMLS method is even larger. The Table 3 and Fig. 1 also show that for the training data with high value outliers, the results for the conjugate gradient method for the LMLS criterion are rather poor when compared with the FLMLS. The explanation for such results is quite simple. As we noticed, in many cases the FMLS algorithm converges

Table 1. The mean MSE for the 500 trials for the networks trained to approximate function of one variable and two-dimensional spiral (Clean Data)

	Function of one variable		Function of two variables	
Algorithm	Mean	S.D.	Mean	S.D.
LMLS	0.0007	0.0001	0.0004	0.0002
FLMLS	0.0014	0.0013	0.0006	0.0003
BP	0.0007	0.0001	0.0003	0.0002

Table 2. The mean MSE for the 500 trials for the networks trained to approximate function of one variable

	Data with gross errors (Type 1)						Data with high value outliers (Type 2)					
	$\delta = 0.1$		$\delta = 0.2$		$\delta = 0.3$		$\delta = 0.1$		$\delta = 0.2$		$\delta = 0.3$	
Algorithm	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
LMLS	0.0061	0.0019	0.0088	0.0035	0.0114	0.0046	0.0050	0.0015	0.0053	0.0016	0.0056	0.0021
FLMLS	0.0054	0.0022	0.0070	0.0030	0.0087	0.0042	0.0043	0.0019	0.0048	0.0018	0.0054	0.0021
BP	0.0398	0.0195	0.0809	0.0351	0.1096	0.0416	1.7929	0.6893	4.0996	1.6899	5.9862	2.5492

Table 3. The mean MSE for the 500 trials for the networks trained to approximate two-dimensional spiral (Data Type 1)

	Data with gross errors (Type 1)						Data with high value outliers (Type 2)					
	$\delta = 0.1$		$\delta = 0.2$		$\delta = 0.3$		$\delta = 0.1$		$\delta = 0.2$		$\delta = 0.3$	
Algorithm	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
LMLS	0.0584	0.0441	0.1442	0.1882	0.0579	0.0331	0.0326	0.0439	0.0140	0.0061	0.0889	0.1575
FLMLS	0.0171	0.0076	0.0259	0.0172	0.0382	0.0208	0.0152	0.0066	0.0161	0.0057	0.0194	0.0089
BP	0.3967	0.2990	0.7722	0.3739	45.5307	12.7130	23.6145	8.6535	36.6818	11.8470	46.2942	12.1067

better than other optimization methods applied for the LMLS error criterion. Lower value of the robust criterion means better performance for the contaminated data and slightly higher error for the clean data. This is why the FLMLS seems to be more robust method than simple LMLS. Simultaneously its performance for the clean data is, as might be seen, acceptable.

In the figure 2 the convergence of tested algorithms was compared. For the clean training data (Fig. 2), the FLMLS algorithm achieves definitely the lowest error criterion (LMLS) level. The error goal obtained after 150 epochs of the conjugate gradient algorithm, the novel FLMLS method achieves in about 10 epochs. For the training data containing gross errors (Fig. 2), the difference between two faster algorithms becomes smaller. However, the LMLS error level is lower for the FLMLS method. Also in this case, the error after 150 epochs of the conjugate gradient is approximately equal to the error after 10 epochs of the FLMLS. It is worth mentioning, that for robust error measures, even slight changes in the obtained error value may have large impact on the final network performance.

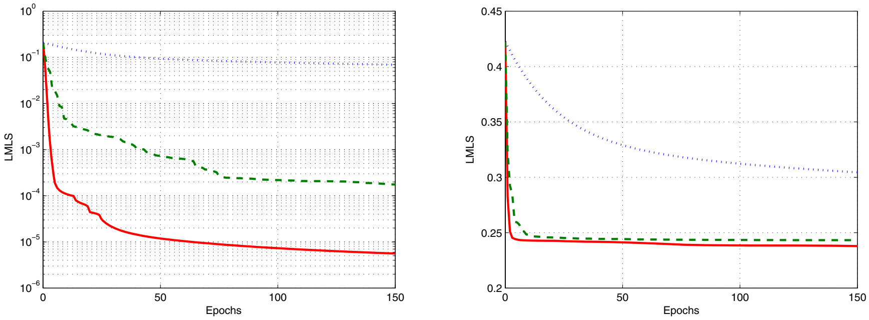


Fig. 2. Convergence of three algorithms with LMLS error criterion (training on the clean data - left, on the data with gross errors, $\delta = 0.2$ - right): gradient-descent algorithm (*dotted line*), conjugate gradient algorithm (*dashed line*), FLMLS algorithm (*solid line*)

4 Conclusion

The novel fast learning algorithm dedicated to the LMLS error criterion was proposed in this article. The idea of the training method was inspired by the

Levenberg-Marquardt algorithm, though, because the error function was not of the quadratic form, a new way of gradient calculation and Hessian approximation was designed. As it was demonstrated by many simulations, the FLMLS not only converges faster than the conjugated gradient method combined with the LMLS criterion, but also acts better in the presence of gross errors in the training data. It may be considered as universal and efficient tool to increase the robustness to outliers. Moreover, it is currently the fastest robust learning algorithm because it uses very simple and well-known modification of the error criterion combined with new optimization method.

References

1. Chen, D.S., Jain, R.C.: A robust back propagation learning algorithm for function approximation. IEEE Transactions on Neural Networks 5, 467–479 (1994)
2. Chuang, C., Su, S., Hsiao, C.: The Annealing Robust Backpropagation (ARBP) Learning Algorithm. IEEE Transactions on Neural Networks 11, 1067–1076 (2000)
3. Hagan, M.T., Demuth, H.B., Beale, M.H.: Neural Network Design. PWS Publishing, Boston (1996)
4. Hagan, M.T., Menhaj, M.B.: Training Feedforward Networks with the Marquardt Algorithm. IEEE Trans. on Neural Networks 5(6) (1994)
5. Hampel, F.R., Ronchetti, E.M., Rousseeuw, P.J., Stahel, W.A.: Robust Statistics the Approach Based on Influence Functions. John Wiley & Sons, New York (1986)
6. Haykin, S.: Neural Networks - A Comprehensive Foundation, 2nd edn. Prentice Hall, NJ (1999)
7. Hornik, K., Stinchconbe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks 2, 359–366 (1989)
8. Huber, P.J.: Robust Statistics. Wiley, New York (1981)
9. Liano, K.: Robust error measure for supervised neural network learning with outliers. IEEE Transactions on Neural Networks 7, 246–250 (1996)
10. Marquardt, D.: An algorithm for least squares estimation of non-linear parameters. J. Soc. Ind. Appl. Math., 431–441 (1963)
11. Olive, D.J., Hawkins, D.M.: Robustifying Robust Estimators, NY (2007)
12. Pernia-Espinoza, A.V., Ordieres-Mere, J.B., Martinez-de-Pison, F.J., Gonzalez-Marcos, A.: TAO-robust backpropagation learning algorithm. Neural Networks 18, 191–204 (2005)
13. Rusiecki, A.L.: Robust MCD-based backpropagation learning algorithm. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 154–163. Springer, Heidelberg (2008)

Using Neural Networks for Simplified Discovery of Some Psychological Phenomena

Ryszard Tadeusiewicz

AGH University of Science and Technology
al. Mickiewicza 30, 30-059 Krakow, Poland
rtad@agh.edu.pl
http://www.tadeusiewicz.pl

Abstract. Neural Networks are often used for solving practical problems and are known and appreciated as an effective soft computing tool. Nevertheless we cannot forget that neural networks are models of parts of the biological brain. Very simplified models, but similar in general behavior to some psychological phenomena. Therefore sometimes we can obtain interesting observations during study of neural network behavior and we can discover on this basis some new psychological ideas. Paper presents an example of such approach. Self-learning process (called also unsupervised learning) is an interesting form of neural network application, slightly different from other neural networks issues. During this process the network ought to discover new knowledge instead of registration of existing knowledge performed by the network during normal supervised learning. This process is described and discussed in many papers, but all of them are goal oriented ones: the main goal of the research is how to obtain the best self learning result e.g., in term of input data classification or similarity measurement. Meanwhile, during the self-learning process some phenomena can be encountered, very interesting from the psychological point of view, when the neural network self-learning process is considered as a model of cognitive processes occurring in our mind during self-learning or thinking. Such phenomena, observed during neural networks self-learning processes are described in the paper. Their psychological interpretation lead to hypothesis of artificial dreams, which can be discovered in neural networks and first time reported in this paper. In the paper, examples of such artificial dreams are presented and discussed. The problem under consideration is definitely controversial one, but the phenomena itself is interesting as a new interpretation of processes observed in neural networks.

1 Introduction

Neural networks are usually regarded only as a tool for intelligent computations. Among these applications neural networks (NN) are known and used also by the people, who definitely are not interested in neurocybernetics, and consider NN only as a computational tool for solving practical problems. Such approach is represented in figure 1, where the common used schema of neural network application is presented in a very condensed form [1]. Most important element in fig. 1, as everybody knows, is knowledge collected in neural network structure during learning process. This element will be considered in further discussions in this paper. Dynamic of the learning processes and

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 104–123, 2010.
© Springer-Verlag Berlin Heidelberg 2010

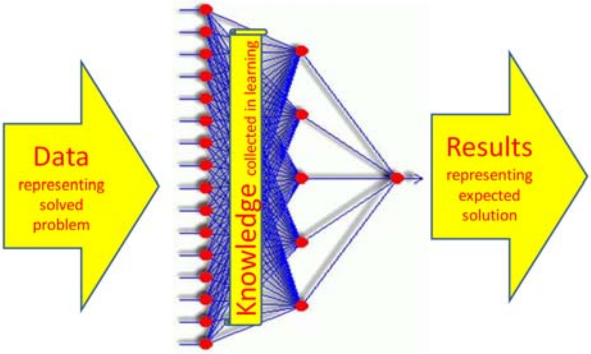


Fig. 1. Typical view of neural network application

unusual phenomena encountered during this process will be one of the most important connections with the psychological phenomena which is mentioned in the title of this paper.

Possibility of the use NN as a model for psychological processes discovery is a consequence of the neural network origin. As can be observed in fig. 2, neural networks are models of some properties of the brain. The way leading to current artificial (technical) neural networks started many years ago from curiosity of the brain researchers (see the element described by letter A in fig. 2). Because of such curiosity many generations of brain researchers have been collecting more and more knowledge about this mysterious organ (B) year by year. Results of biological investigations can be transformed - by biocybernetic researchers (C) - into mathematical models (D), where such models programmed for digital computers can be used as widely known neural networks (E). This path is well known. But there is a question: Are the results of the neural networks observations useful for understanding of the real brain mechanisms? This element is represented in Fig. 2 as the arrow with question mark.

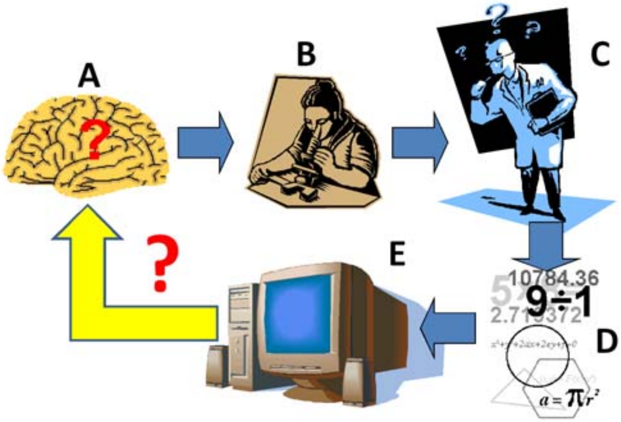


Fig. 2. The way of neural networks development and possible use neural networks as a models of brain functioning and psychological processes as a result of such its biological activity

The main goal of this paper is to show arguments for removing such question mark and for acceptance of the truth, that neural networks as models of parts of biological brain can be useful also for discovery of the brain properties.

Somebody can claim, that neural networks are very simplified models of the real biological neural systems. It is obviously truth. In fact, the relation between complexity of the brain and simplicity of typical artificial neural network is the same, as the relation between mass of the Earth and mass (or volume) of a pinhead (Fig. 3)! Is there a chance (hope?) that so much reduced and simplified model can really help us to understand processes in the whole complex brain?

The answer is positive. In scientific discovery volume is not important factor. We must take into account such circumstance: If something is proven in a small drop, it works also in huge ocean (Fig. 4). Moreover, in complex system, discovery simplicity

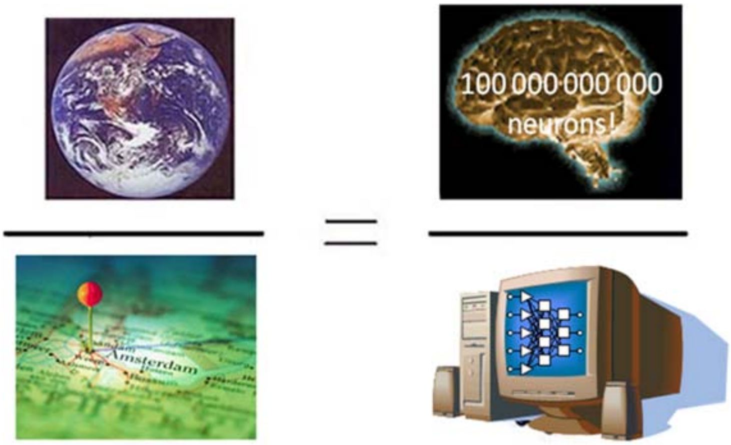


Fig. 3. Relation between complexity of the brain and simplicity of a neural network

Fig. 4. In scientific discovery dimension is not so important. Rules proven in a small drop work also in huge ocean.

of a model used can be in fact main advantage (Fig. 5) because we cannot understand whole brain as a complex, multi-element and multi-function system. But simple models, which are built on the base of neural networks, can give us understanding of everything without difficulty. There is also another argument for use neural networks for interpretation of brain research results. Nowadays, science brings us a lot of information about the brain structure and functioning through many kinds of investigations (fig. 6). But in this model of knowledge development, every collected result must be considered separately: in this place we have a collection of morphological observations, in another places electrophysiology, biochemical results, clinical observations... Everything is complete but describes only a part of the knowledge about the brain. But who and where does reconstruct the whole? Such separate collections of particular data can be integrated only on the base of a holistic model. Until we are not able to build better model of the brain

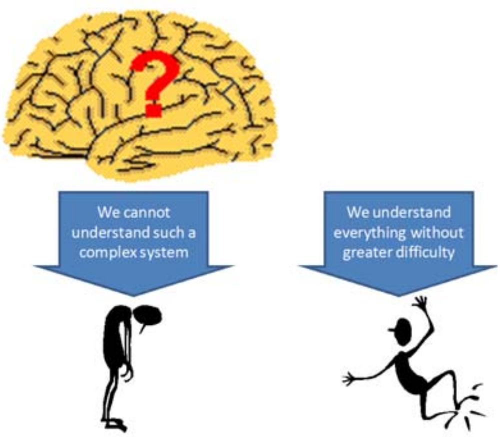

We cannot understand such a complex system
We understand everything without greater difficulty

Fig. 5. We can investigate and understand processes in neural networks whereas understanding the same processes in the brain is out of our possibilities

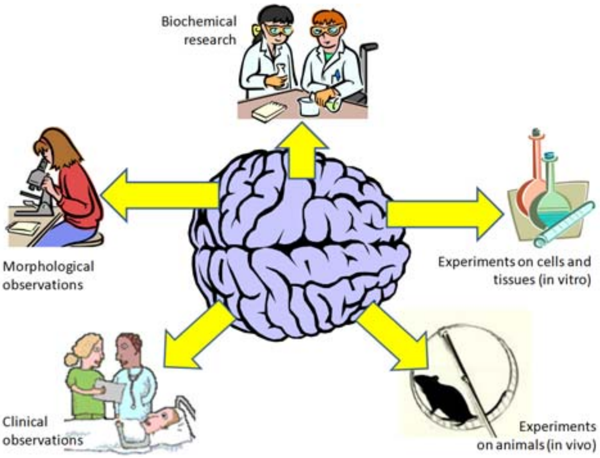

Biochemical research
Morphological observations
Experiments on cells and tissues (in vitro)
Clinical observations
Experiments on animals (in vivo)

Fig. 6. Particular branches of science collect knowledge about the brain separately

– neural networks could not serve as a framework for integration of data, which are derived from many types of research. Of course, in the case of pursue toward the whole brain model.

The problem under consideration is generally very complicated, but we can show in this paper example, how to use neural networks for simulating very simple systems, similar in general behavior to some psychological phenomena. The system presented in this paper is reduced to some simple elements working on the base of very simplified set of rules, but nevertheless we can obtain interesting observations during the study which suggest that learning neural network behavior allows us to discover some new psychological ideas.

2 General Presentation of the Idea

This paper is focused on phenomena, occurring during neural network self-learning process (called also unsupervised learning [3]), which still are not yet fully characterized and understood. It seems that machine learning and self-learning algorithms are already well known and a possibility of new interesting scientific discoveries in this area is very small. Such a statement is true, but only if we consider learning as a goal oriented optimization processes. If we return to the most basic definition of learning, as a knowledge building process both in a human mind and in machines – we can observe interesting and not yet discovered phenomena. It seems that the most interesting observations are related to the self-learning process. Because of a similarity to the psychological processes observed in the biological brain we call such phenomena ”artificial dreams” [4]. This name is similar to the title of Hamid Ekbia’s book [2], but the meaning of this term in our paper is slightly different. In Ekbia’s book ”artificial dreams” are presented as unrealized and unrealizable projects related to the Artificial Intelligence. In our research we observe the ”artificial dreams” as spontaneous and unexpected processes, automatically emerging from the natural self learning procedures.

The phenomena under consideration are very interesting and exciting. Therefore it is quite surprising, why they are so rarely reported by the Artificial Intelligence researchers? If so many people perform self-learning processes for many purposes – why such phenomena are still not discovered and described?

The answer is simple. Most papers describing methods and results of the self-learning (even in neural networks, which are considered in this work) are definitely goal-oriented. Researchers or practitioners are concentrated on the applications, not on the tool and its behavior. First, authors of almost all papers try to obtain the best result in terms of solving specific problems (e.g., building of neural network based model of some processes or finding a neural solution of a pattern recognition problem). For this purpose researchers think about problem statement, best network architecture selection, fast learning for receiving results and detail analysis of these results (Fig. 7). Almost nobody tries to study learning processes not as the way to the goal, but as a goal itself. These research, dealing with learning and self-learning itself, are also mainly dedicated to the speeding of the learning process or to increasing the quality of final results (e.g. in terms of avoiding the local minima problem). They usually do not take into account what can happen in the network **during the learning process**.

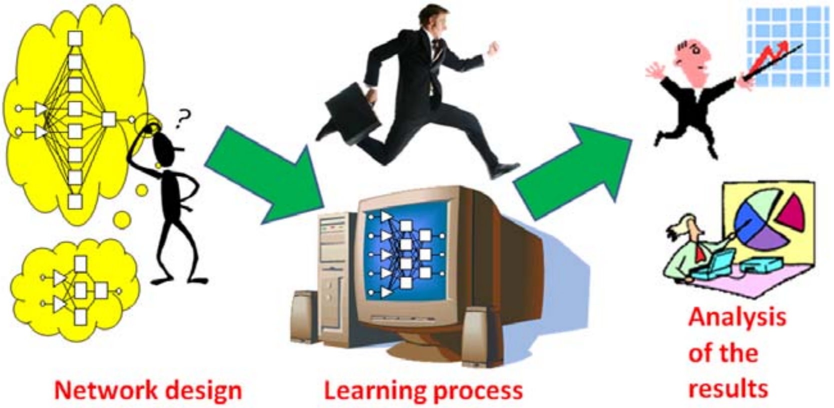


Fig. 7. Using of neural networks in goal-oriented works. Nobody has time for careful observation of transitory processes during network learning.

In this paper we try to take into account such neglected phenomena observed during the self-learning process and we try to provide a psychological interpretation to such phenomena. The unexpected outcome of our research seems to be really surprising: we can look at amazing process of machine dreaming. . . .

3 Self-learning and Learning (or Supervised and Unsupervised Learning)

In this paper we take into consideration self-learning processes, instead of better known and more useful (from technical point of view) machine learning processes. Understanding of this difference will be crucial to the main thesis of this paper. Short additional remark of these issues is done in figures 8 and 9 on the base of a gender recognition problem. It is obvious that the problem under consideration is known for most readers, but the figures can be helpful for readers not familiarized with machine learning.

During the regular learning process we have the ”teacher”, who teaches a ”pupil” (in fact it is a machine) on the base of examples of properly solved tasks. In the machine learning the teacher is an algorithm, powered with examples database, but the main idea of teaching is based on a simple scheme: get the knowledge from the teacher and put it in the pupil. After learning process, the ”artificial pupil” can take an exam, where the quality of learned knowledge can be evaluated and assessed. Fig. 8 presents this idea. In contrast to the previous scheme, the self-learning process is based on the ”knowledge discovery” methods. The pupil (in fact it is still a machine) can accept input data, but there is no teacher who can explain what the information means. Therefore, self-learning not only has to accumulate knowledge, but it must **discover** this knowledge without any external help. In general, it is a difficult task, but many successful applications prove this way to be effective.

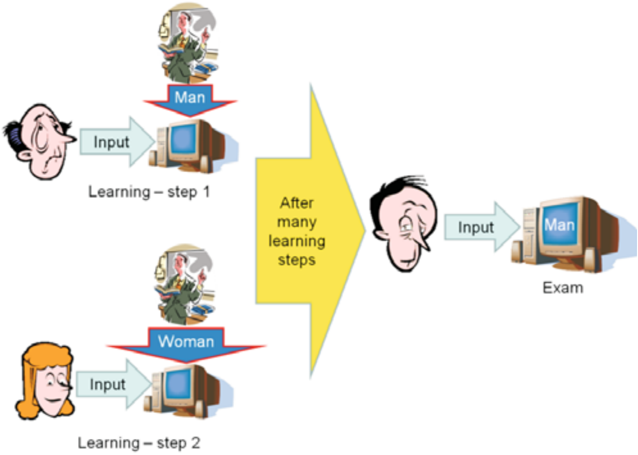


Fig. 8. Learning and exam in the supervised learning

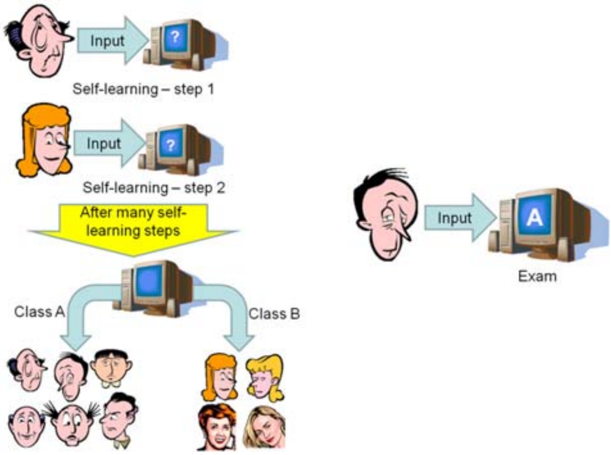


Fig. 9. Learning and exam in unsupervised learning (self-learning)

The self-learning process on the base of gender recognition problem is presented in Fig. 9. The self-learning system after connecting with many input data can differentiate men from women, but of course it cannot give the proper names for the genders. During the exam, the self-learning system can classify a new person (sometimes properly, and sometimes not – as every artificial classification system) using symbols of classes instead of names.

The question is:

Why and where unsupervised learning systems are necessary?

Let us take into account unmanned distant and not known planet. The computer located on board must register and classify the extraterrestrial aliens which will be observed by spaceship sensors, when the researchers cannot forecast in advance, which and how many species of such aliens could be registered (Fig. 10). In the situation described above the best solution is to use a self-learning system, which could register observed aliens, collect it according to their similarity and discover on this base new species and new classification methods, absolutely not predictable in advance. Content of the memory of the self-learning spacecraft main computer after discovery of an alien planet is shown in Fig. 11. This presents useful and nice self-learning process results. Let us consider that in this hypothetical situation, the application of self-learning system is the only possible solution and in fact can be very effective. There are many learning and self-learning systems, but for purpose of this paper we selected neural networks [5] as a tool, in which the ”artificial dreams” phenomena are observed [2]. General concepts of neural networks are well-known. However, we will try to tell more about a simple (and interesting!) application of a self-learning neural network, which will be a base for further discussion.

Fig. 10. Hypothetical application of a self learning system. Discussion in the text.

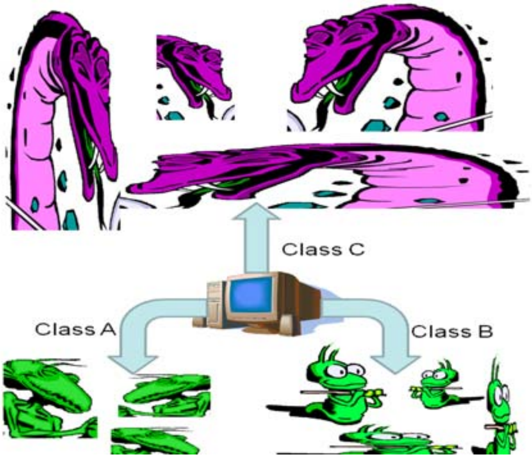


Fig. 11. Content of the memory of a self-learning spacecraft after alien planet discovery

4 Considered Neural Network

The phenomena described in this paper can be discovered, as mentioned above, in almost all types of neural networks and for almost all methods of learning (both supervised and unsupervised). In this paper we decided to take into account the following situation: Let us have a one-layer linear neural network (Fig. 12). It means that n-dimensional vector $\mathbf{X} =< x_1, x_2, \dots, x_n >$ is passed as the input to the network and is processed using the knowledge of the network which is represented by a collection of weight vectors $\mathbf{W}_j =< w_{1j}, w_{2j}, \dots, w_{nj} >$ for all neurons ($j = 1, 2, \dots, k$).

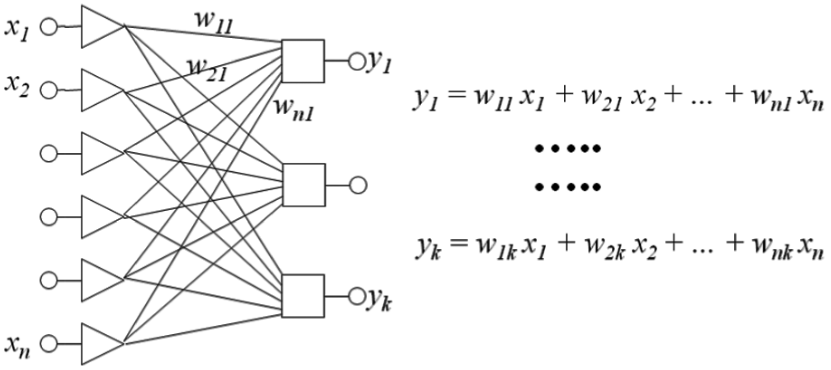

x_1
x_2
x_n
w_{11}
w_{21}
w_{n1}
y_1
y_k
$y_1 = w_{11} x_1 + w_{21} x_2 + \dots + w_{n1} x_n$
$y_k = w_{1k} x_1 + w_{2k} x_2 + \dots + w_{nk} x_n$

Fig. 12. Structure of a neural network in which self-learning processes can be observed

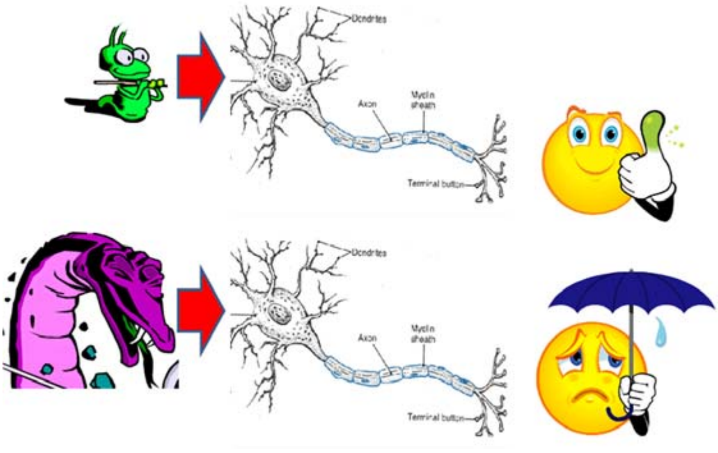


Fig. 13. For some input vectors, the answer y_i given by the neuron is positive and for others could be negative

The output y_i produced by each neural node can be obtained by means of the simplest and well known equation:

$$y_j = \sum_{i=1}^{n} w_{ij} x_i \tag{1}$$

As everybody knows, this equation causes differentiation between objects described by vectors $\mathbf{X} =< x_1, x_2, \ldots, x_n >$. For certain input vector $\mathbf{X}_A$ the answer y_i given by the neuron is positive and for other $\mathbf{X}_B$ may be negative (Fig. 13).

It is absolutely clear from equation (1) that the neuron answer depends on weight vector $\mathbf{W}_j =< w_{1j}, w_{2j}, \ldots, w_{nj} >$. Taking into account different weight distributions we can expect different neuron answers for the same input (Fig. 14).

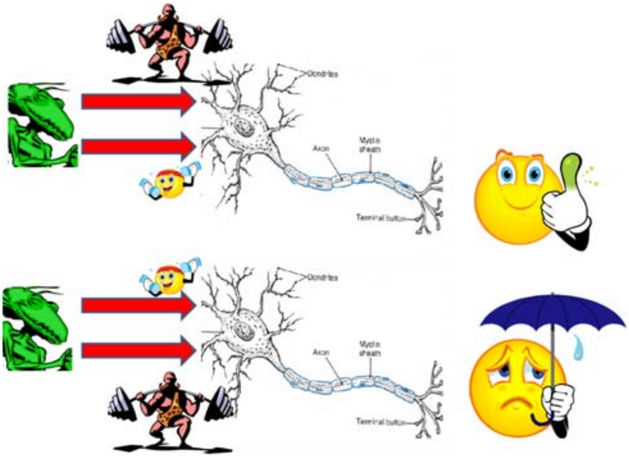


Fig. 14. For the same object the answer of a neuron can vary depending on weights assigned to particular components of input vector

5 Learning of Considered Neural Network

The network learns on the base of a simple Hebbian rule (Fig. 15): If on step p one obtains the input vector $\mathbf{X}_p =< x_{1p}, x_{2p}, \ldots, x_{np} >$ than the correction of the weight vector $\Delta\mathbf{W}_j(p)$ depends on the output value y_{jp} calculated by the j-th neuron for $\mathbf{X}_p$ according to the equation (1), and on the value of input vector $\mathbf{X}_p$ according to the formula

$$\Delta\mathbf{W}_j(p) = \eta y_{jp} \mathbf{X}_p \tag{2}$$

where η is the learning rate coefficient ($\eta < 1$). Of course a new value of the weight vector $\mathbf{W}_j$ at the next step $(p+1)$ of the self-learning process can be calculated by means of formula:

$$\mathbf{W}_j(p+1) = \mathbf{W}_j(p) + \delta\mathbf{W}_j(p) = \mathbf{W}_j(p) + \eta y_{jp} \mathbf{X}_p \tag{3}$$

which must be applied for all neurons (for all $j = 1, 2, \ldots, k$). It is easy to find out, that the result of such calculations are different for neurons with positive output y_{jp}

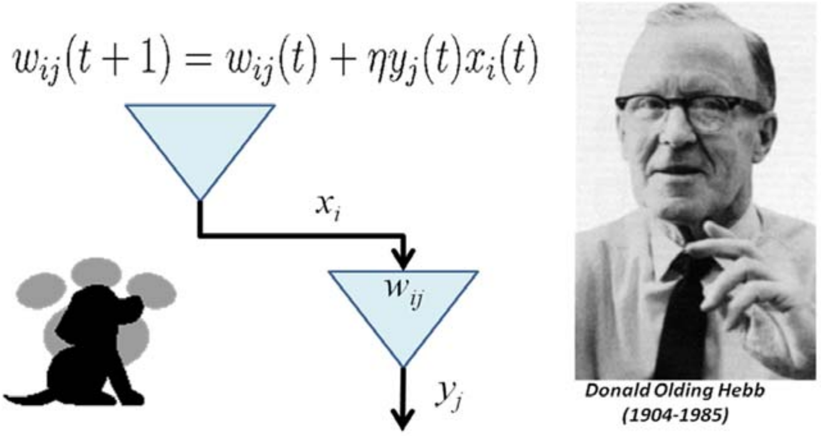


Fig. 15. Foundations of self-learning are based on Hebb's research on learning processes in animal brains

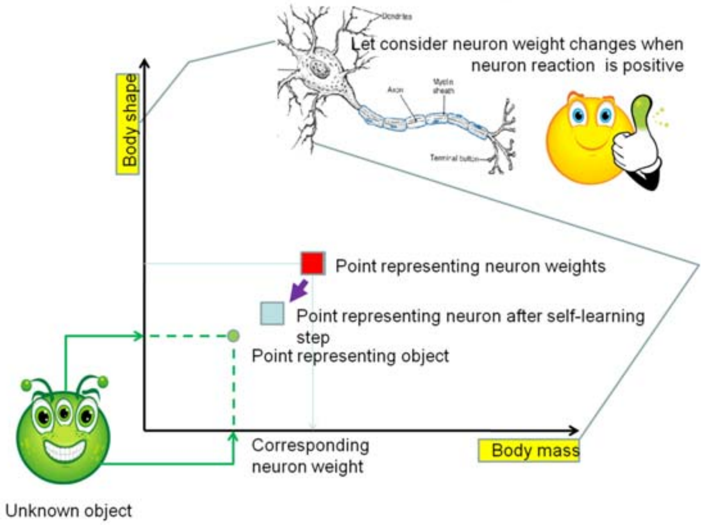

Dendrites
Let consider neuron weight changes when
neuron reaction is positive
Axon
Myelin sheath
Terminal button
Body shape
Point representing neuron weights
Point representing neuron after self-learning step
Point representing object
Corresponding neuron weight
Body mass
Unknown object

Fig. 16. When output of neuron for given input signal is positive the result of the self-learning step is the attraction of the neuron weight vector toward the input signal position

calculated as the answer for input signal $\mathbf{X}_p$ (Fig. 16), and different for neurons with negative output. In the first case the weight vector of the neuron $\mathbf{W}_j(p)$ is changed toward the position of the actual input signal $\mathbf{X}_p$ (attraction), while in the second case the weight vector of the neuron $\mathbf{W}_j(p)$ is changed backward to the position of the actual input signal $\mathbf{X}_p$ (repulsion – Fig. 17). This process for big set of neurons is presented on Fig. 18, where a big green ring denotes position of the input signal $\mathbf{X}_p$ and small squares denote positions of weight vectors of the neurons. The ”migration” of the

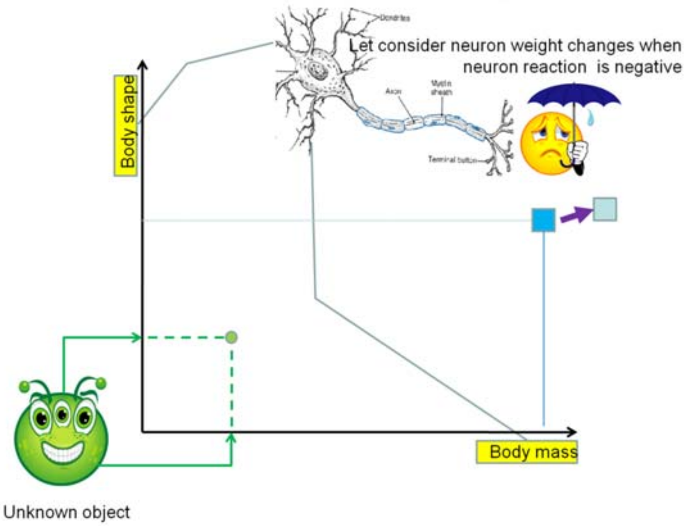


Fig. 17. When the output of neuron for given input signal is negative the result of the self-learning step is the repulsion of the neuron weight vector toward input signal position

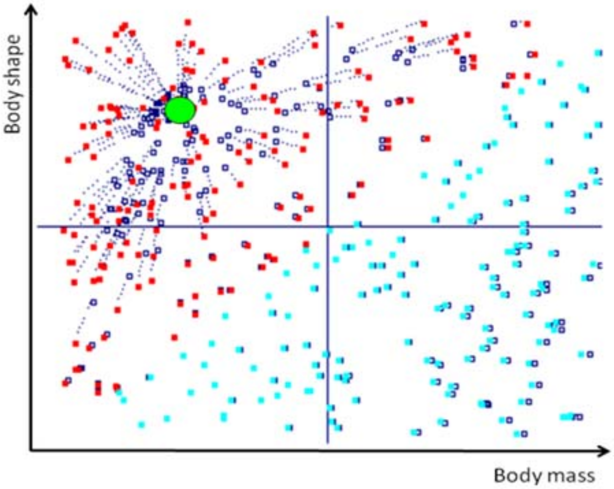


Fig. 18. Migration of the weight vectors in the biggest self-learned network

weight vectors can be observed on this plot – some are attracted toward the input signal, whereas the other are pushed in the opposite direction. Initial position of the weight vectors of neurons with positive output are plotted as red points whereas weight vectors of neurons with negative output are plotted as light blue points. The final position of the weight vectors for both type neurons are marked dark blue. One can see that the red points are attracted toward the input signal position, where the other are pushed (slightly) in the opposite direction (See also comments to Fig. 16 and 17). It is well

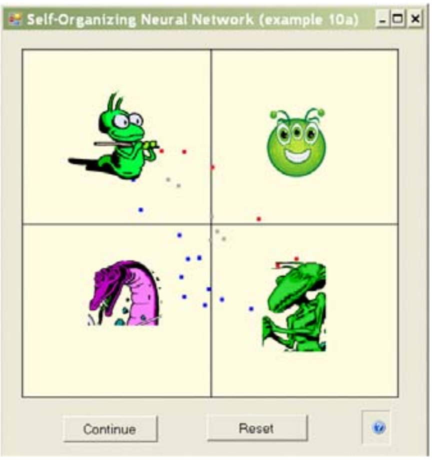


Fig. 19. Different objects are located in different regions in neuron weight space

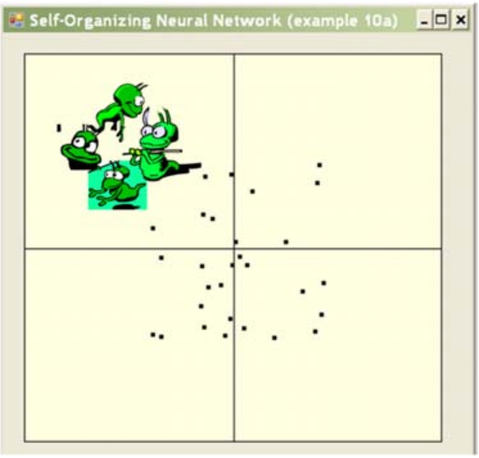


Fig. 20. Similar objects are located in a compact region

known what is the result of such a self learning process after many steps performed by the network connected to a real data stream. If the data are not uniformly distributed, the neurons are divided (spontaneously!) onto groups, when every group is dedicated to the one cluster of the input data. It can be observed on the result window presented by the program developed by the author for research of the self learning process in simple neural networks (Fig. 19). Let me remark, that both plot for Fig. 18 and the next figures were produced using the same program. The program itself is available (free) on the page http://home.agh.edu.pl/˜tad//index.php?page=programy&lang=en . Moreover, the values of the weight vectors of the neurons belonging to each group are more or less focused and located in the center of selected cluster of the data (Fig. 20). It means, that after the self-learning process there are neurons inside the neural network, which can be used as detectors (or sentinels) for every cluster (group of similar signals), present in the observed data stream and that they can be automatically discovered by the network.

6 How It Works after Learning Process?

When the learning process is finished, every neuron can act as a detector of some type of input object. Let us consider the network learned using many images of women (Fig. 21). After learning process, when a new woman image appears at last one neuron from the women-trained group gives large positive answer (output signal), what can be interpreted as recognition of such a not known object as belonging to the class of women (Fig. 22). Self learning neural network can be considered as a self formatting classification tool, which can be used for the recognition of new objects and their categorization as belonging to classes which are not known *a priori*.

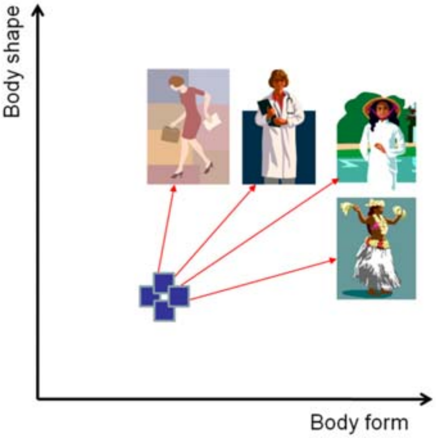


Fig. 21. After learning process, different neurons can recognize different women images, but they are grouped together in a small subarea of the input signal space

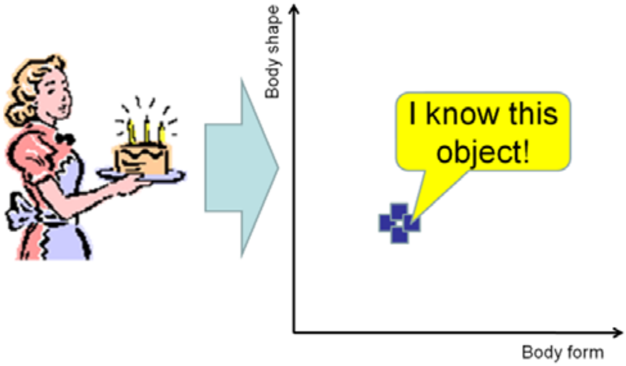


Fig. 22. Learned network can recognize new objects belonging to the given classes

The process described above is not ideal, because, as everybody knows, spontaneous migration of the weight vector for every independent neuron leads to many pathologies, such as: i) when every attractor has many neurons as the detectors (over-representation), and ii) when some important attractors can be omitted (none of neurons decides to point out this region of input space). Everybody knows also how to solve such a problem: the much better solution is to use the Kohonen network and methodology of self-organizing maps.

However in this work we do not try to make the best self-organized representation of the data. Our goal is entirely different: we are looking for a very simple model of learning of neural network, because on the base of such a model we are going to show, how (and why) the trained network sometimes exhibits behavior, which can be interpreted as the ”artificial dreams”.

7 How and Where Artificial Dreams Phenomena Can Be Discovered?

Let us see Fig. 23. The starting point of the self learning process is known (chaotic distribution of weights in all considered neurons). The ending point after learning is also known: we observe clusters of neurons, which are sentinels of particular classes discovered after spontaneous cauterization performed during self-learning process. But which interpretation can be bound with characteristic conformation of weights coefficients (Fig. 23), observed in transition states during the self learning process? Such phenomena can be discovered long time after the start of learning, when the network knows nothing because of random values assigned to all its weights. Self-learning process goes then automatically, so typical researcher starts to uptake another job or goes home. At the same time, considered in this paper ”unusual phenomena” can be observed long time before a final point of learning process is reached, i.e. when the network knows (almost) everything and can be exploited according to the plan. Such phenomena can be

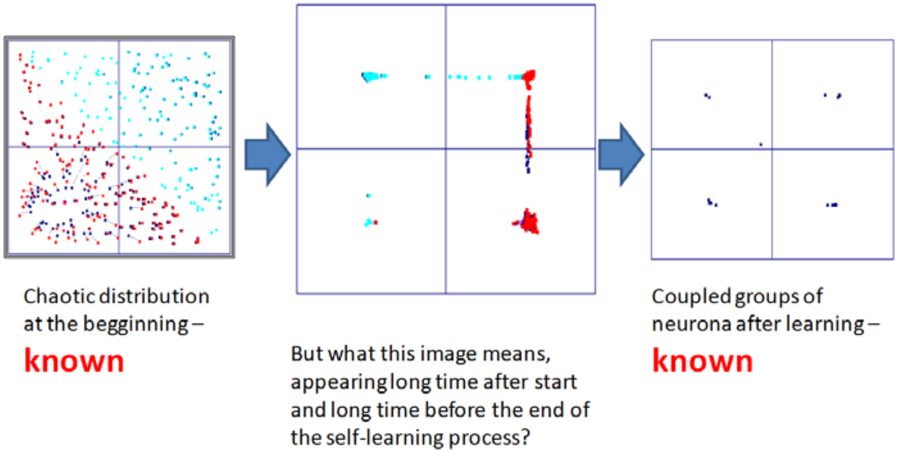

Chaotic distribution
at the begginning –
known
But what this image means,
appearing long time after start
and long time before the end of
the self-learning process?
Coupled groups of
neurona after learning –
known

Fig. 23. When self-learning is in progress – we can observe something strange. . .

classified as simple errors or more subtle imperfections of not matured enough self-learning neural network and than can be disregarded. But such phenomena can be studied and found interesting, when we can give them some psychological interpretations.

8 How Artificial Dreams Are Manifested?

This observed phenomena can be disregarded as learning imperfections, but some of them can also be interpreted as ”artificial dreams” performed by artificial neural networks. It can give us a new interpretation of human ability of imagination, fantasy and also poetry. It can be presented even on the base of very simple neural network models, but of course the most interesting results can be investigated by means of the networks deployed with a high level of similarity to the real brain structures. That means a huge level of complication of the neural structure and also complicated forms of observed phenomena. Before considered phenomena are shown and discussed, a short description of the example problem needs to be provided, in which ”artificial dreams” can be very easy encountered.

Let us take a very simple problem, which should be solved by the neural network during the self-learning process. There are four clusters in the input data. Let assume for clear and easy graphical presentation of the results, that the attractors preset in the data (most typical examples) are localized exactly at the centers of four subparts (quarter) of the input space (Fig. 24). The base of this space is defined by two parameters: *body form* and *body shape* (whatever it means). In such space one can observe a process of differentiation of four various groups of living beings (women, birds, fishes and snakes) shown (one example for every class) in Fig. 24. In this case, the self-learning process in the simulated neural network after some thousands of learning steps leads to the

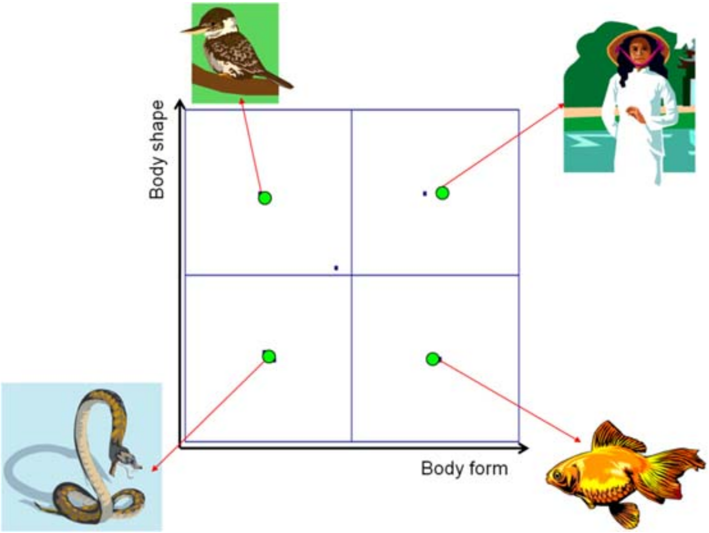


Fig. 24. The example problem. Detailed description in the text.

situation, when almost every neuron becomes a member of one of the four separate groups, located (in the sense of localization of the weight vectors) at the points corresponding with the centers of the clusters discovered in the input data stream. Three snapshots from the learning process are presented in the Fig. 23.

Typical user of the neural network takes into account mainly last snapshot, presenting, how many neurons are located in the proper positions after the learning process and how precisely the real values of attractors coordinates are reproduced by the neurons parameters. For our consideration the middle snapshot will be the most interesting, because it presents something strange: a situation, when knowledge of the network is definitely not complete, but also the initial chaos was partially removed. This stage of learning process is usually skipped by neural network researchers, because apparently one cannot find anything interesting in this plots: the learning process is not ready yet, that is all.

In fact, what we see on the central plot of Fig. 23 is an evidence of ”artificial dream” appearance. We must only think in terms of special interpretation…

9 Special Interpretation of the Intermediate Stages of the Learning Process

In all goal-oriented investigations of neural networks researchers are interested in the final result of learning process, which must be useful and accurate. Almost nobody takes into consideration intermediate stages shown in Fig. 23. However, if one tries to provide interpretation to the form of the plotting, repeated in Fig. 25 – one may say that although it is not a real **dream**, it can be interpreted as a very exciting **model of the artificial dream**. In fact, on the plotting presented in Fig. 25 we can point out the localizations of the neurons, which can recognize some (named) objects from a real world. After training of all neurons they can be attributed to the real world objects, like girls, fishes and birds. However, on a very early stage of the learning process, one can find in the population of neurons both real-world related detectors and fantasy-world related detectors. On the line connecting points representing for example girls with the point representing fishes we can find neurons, which are ready to recognize objects, which parameters (features) are partially similar to the girls shapes, and partially include features taken from the other real objects, for example fishes. Another hybrid imagination is a creature having features taken from girls and birds. Perhaps it can be an angel? Isn’t it something like ”dreams” observed in the plots shown in Fig. 25? Obviously in the real world, some objects like that plotted here cannot exist. The objects of such weird properties cannot be the element of the learning data stream, because the input information for the network is taken every time from the real world examples. Nevertheless, in the neural network structure, the learning process formed neurons, which want to observe and recognize such not-so-real objects.

Isn’t it some kind of the ”artificial dreams”?

Very interesting is a fact, that the fantasy-oriented objects, like presented in Fig. 25, encountered during the learning process, are never unrestricted or simply random. It is only possible to find such neurons, which are able to recognize some fantastic hybrids, but build from the real elements. It seems to be an analogy to stories or myths.

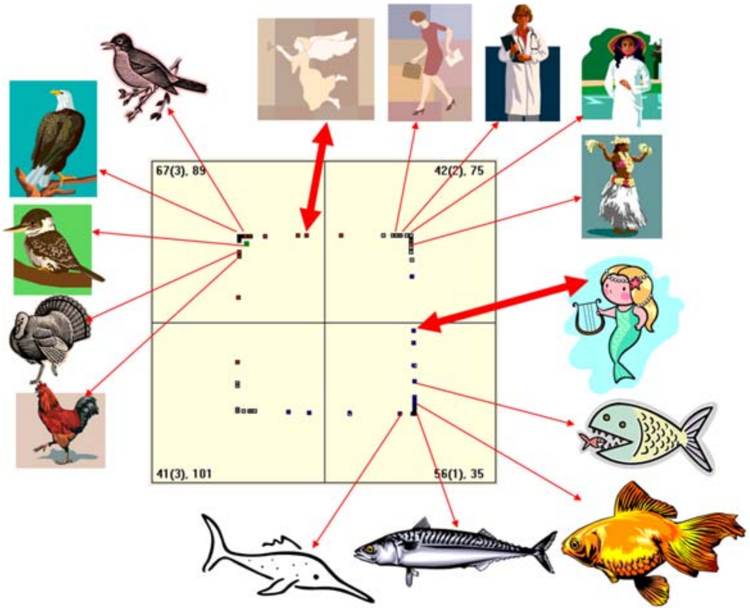


Fig. 25. Parameters of a self learning neural network show after encoding, that some of neurons spontaneously produce imaginations of non existing beings. There are the ”artificial dreams”!

Due to a limited volume of this paper it is not possible to provide many other examples of the ”artificial dreams” encountered during neural networks learning processes. Nevertheless, one more example can be also beneficial and interesting, as it shows another kind of fantasy identified in the neural network behavior. This form of fantasy can be called ”giants making”. Example of such behavior of the trained network is presented in the Fig. 26. When the network is trained by means of examples from real world objects – in the neural structures the prototypes of such objects are formed and enhanced. This process goes over the big population of neurons and leads to the formation of the internal representation (in neural structures) of particular real objects. Neurons belonging to these representations can recognize every real object of the type under consideration. It is a well known and regular process. But sometimes, in contrast to this regular pattern, we can observe single neurons, which parameters are formed in such way, leading to a surprise after interpretation. Let us assume, that the network was trained based on ”lions” as real objects (Fig. 26). The network can ”see” many lions (of course as a collections of parameters, representing selected data about lions – e.g. how tall is lion, how long and sharp are lion’s teeth and so on). After some learning period, inside the network there is some kind of an imagination of the real lion. This imagination, given as collection of parameters (neurons weights), enable us to recognize every real lion. But some neurons have parameters, which enable to recognize a surreal lion, much bigger than the real one, with bigger teeth and with much more dangerous claws. The relations and proportions between parameters are the same, as for real lions (see Fig. 10 for relations between parameters of real objects and relations between parameters of the imprinted in weights of refugee neuron imagination of the ”giant” – both

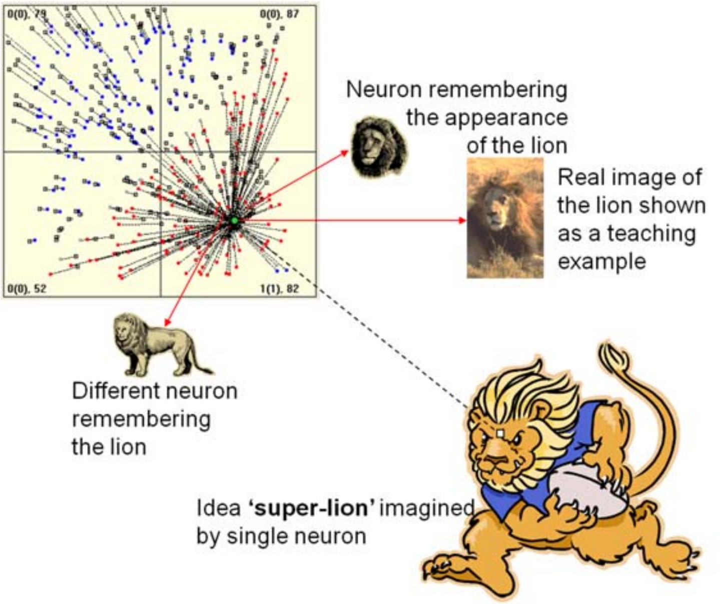

0(0). 79
0(0). 87
0(0). 52
1(1). 82
Neuron remembering the appearance of the lion
Real image of the lion shown as a teaching example
Different neuron remembering the lion
Idea **'super-lion'** imagined by single neuron

Fig. 26. Another form of an "artificial dream". Description in the text

belonging to the same line, coming from the root of coordination system), but such a big lion cannot exist. Nevertheless, one can find neurons ready for recognition of such a giant, although it does not exist!

10 Concluding Remarks

Facts and comments presented in this paper definitely are not so very important from the scientific point of view. They are also not applicable to the practical problem solving using neural networks. However, as long as we use neural networks as the artificial systems very similar to the structures discovered in the human brain, we still think about analogies between processes in our minds and in neurocomputers. Results of simulations presented in this paper gives us a new perspective to such considerations and we hope can be interesting for many neural network researchers bored with new learning paradigms, new network structures and new neurocomputing applications and looking for something absolutely different from the serious and boring standards. This paper is something for them!

References

1. Rutkowski, L., Tadeusiewicz, R. (eds.): Neural Networks and Soft Computing. Polish Neural Network Society, Zakopane (2000)
2. Ekbia, H.: Artificial Dreams: The Quest for Non-Biological Intelligence. Cambridge University Press, Cambridge (2008)

3. Hinton, G., Sejnowski, T.J. (eds.): Unsupervised Learning: Foundations of Neural Computation. MIT Press, Cambridge (1999)
4. Tadeusiewicz, R., Izworski, A.: Learning in Neural Network - Unusual Effects of "Artificial Dreams". In: King, I., Wang, J., Chan, L.-W., Wang, D. (eds.) ICONIP 2006, Part I. LNCS, vol. 4232, pp. 211–218. Springer, Heidelberg (2006)
5. Tadeusiewicz, R.: Computer modeling of self-learning processes in neural network with some psychological remarks. In: Makikawa, M., Wojcicki, J.M. (eds.) New Trends in Biomedical and Clinical Engineering. Proceedings of 104th ICB Seminar, 10th Polish - Japanese Seminar, International Centre of Biocybernetics, Warsaw, p. 43 (2009)

Hybrid Learning of Regularization Neural Networks

Petra Vidnerová and Roman Neruda

Institute of Computer Science
Academy of Sciences of the Czech Republic
Pod vodárenskou věží 2, Prague 8, Czech Republic
petra@cs.cas.cz

Abstract. Regularization theory presents a sound framework to solving supervised learning problems. However, the regularization networks have a large size corresponding to the size of training data. In this work we study a relationship between network complexity, i.e. number of hidden units, and approximation and generalization ability. We propose an incremental hybrid learning algorithm that produces smaller networks with performance similar to original regularization networks.

1 Introduction

The problem of *supervised learning* is extensively studied both in theory and applications. Consider the situation when we are given a set of examples $\{(\boldsymbol{x}_i, y_i) \in R^d \times R\}_{i=1}^N$ obtained by random sampling of some real function f, generally in presence of noise. To this set we refer as *a training set*. The goal is to recover the function f from data, or find the best estimate of it. It is not necessary that the function exactly interpolates all the given data points, but we need a function with good generalization. That is a function that gives relevant outputs also for the data not included in the training set.

In Section 2 we will study the problem of learning from examples as a function approximation problem and show how the regularization network (RN) is derived from regularization theory. In Section 3 we discuss the relation between the network complexity (number of hidden units) and its approximation and generalization ability. We demonstrate the relation between the number of units and training and testing errors on experiments, and derive several recommendations for choosing number of units. Based on these recommendations, we introduce in Section 4 the hybrid learning algorithm combining global search by genetic algorithm and a local search by a gradient descent method. The algorithm is demonstrated on experiments in Section 5.

2 Approximation via Regularization Network

We are given a set of examples $\{(\boldsymbol{x}_i, y_i) \in R^d \times R\}_{i=1}^N$ obtained by random sampling of some real function f and we would like to find this function. Since this problem is ill-posed, we have to add some a priori knowledge about the function. We usually assume that the function is *smooth*, in the sense that two similar inputs corresponds to two similar outputs and the function does not oscillate too much. This is the main idea of

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 124–131, 2010.
© Springer-Verlag Berlin Heidelberg 2010

the regularization theory, where the solution is found by minimizing the functional (1) containing both the data and smoothness information.

$$H[f] = \frac{1}{N}\sum_{i=1}^{N}(f(\boldsymbol{x}_i) - y_i)^2 + \gamma\Phi[f], \tag{1}$$

where Φ is called a *stabilizer* and $\gamma > 0$ is *the regularization parameter* controlling the trade-off between the closeness to data and the smoothness of the solution. The regularization scheme (1) was first introduced by Tikhonov [1] and therefore it is called a Tikhonov regularization. The regularization approach has good theoretical background, it was shown that for a wide class of stabilizers the solution has a form of feed-forward neural network with one hidden layer, called *regularization network*, and that different types of stabilizers lead to different types of regularization networks [2,3].

Poggio and Smale in [3] proposed a learning algorithm (Alg. 2.1) derived from the regularization scheme (1). They choose the hypothesis space as a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}_K$ defined by an explicitly chosen, symmetric, positive-definite kernel function $K_{\boldsymbol{x}}(\boldsymbol{x}') = K(\boldsymbol{x}, \boldsymbol{x}')$. The stabilizer is defined by means of norm in $\mathcal{H}_K$, so the problem is formulated as follows:

$$\min_{f \in \mathcal{H}_K} H[f], \text{where } H[f] = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(\boldsymbol{x}_i))^2 + \gamma ||f||_K^2. \tag{2}$$

The solution of minimization (2) is unique and has the form

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} w_i K_{\boldsymbol{x}_i}(\boldsymbol{x}), \qquad (N\gamma I + K)\boldsymbol{w} = \boldsymbol{y}, \tag{3}$$

where I is the identity matrix, K is the matrix $K_{i,j} = K(\boldsymbol{x_i}, \boldsymbol{x_j})$, and $\boldsymbol{y} = (y_1, \ldots, y_N)$.

```
Input: Data set $\{\boldsymbol{x}_i, y_i\}_{i=1}^N \subseteq X \times Y$                    Output: Function $f$.

1. Choose a symmetric, positive-definite function $K_{\boldsymbol{x}}(\boldsymbol{x}')$, continuous on $X \times X$.
2. Create $f : X \to Y$ as   $f(\boldsymbol{x}) = \sum_{i=1}^N c_i K_{\boldsymbol{x}_i}(\boldsymbol{x})$ and compute $\boldsymbol{w} = (w_1, \ldots, w_N)$ by
   solving
                         $(N\gamma I + K)\boldsymbol{w} = \boldsymbol{y}$,                              (4)
   where $I$ is the identity matrix, $K_{i,j} = K(\boldsymbol{x_i}, \boldsymbol{x_j})$, and $\boldsymbol{y} = (y_1, \ldots, y_N)$, $\gamma > 0$.
```

Algorithm 2.1

The solution (3) can be represented by a neural network with one hidden layer and output linear layer. The most commonly used kernel function is Gaussian $K(\boldsymbol{x}, \boldsymbol{x}') = e^{-\left(\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|}{b}\right)^2}$.

The power of the Alg. 2.1 is in its simplicity and effectiveness. However, its real performance depends significantly on the choice of parameter γ and kernel function type. Optimal choice of these parameters depends on a particular data set and there is no general heuristics for setting them.

3 On the Number of Hidden Units

The solution derived in the previous section contains as many hidden units as is the number of data samples. Such solution is unfeasible for most real life problems. Therefore solutions with lower number of hidden units are considered.

There are two ways how to create such a network of smaller size. The first one starts with the solution obtained by the Algorithm 2.1 and prunes the network. The second approach tries to build smaller networks from scratch.

Our experience suggests that the former approach is not very efficient for RN networks. Table 1 lists the number of units with weights higher than certain threshold, which is 0.001 in our case. Let us consider that the units with weights than the threshold are adepts for pruning, while the rest can be seen as *relevant units*. We can see that only small fraction (in case of the glass task even none) of units can be directly pruned. This makes the first approach rather unfeasible.

Table 1. Number of hidden units with weights higher than 0.001

Task	Relevant units	Total units
cancer1	513	525
cancer2	499	525
cancer3	507	525
glass1	161	161
glass2	161	161
glass3	161	161
heart1	669	690
heart2	659	690
heart3	663	690

Concerning the latter approach, we are interested in the relationship between the number of hidden units and approximation accuracy. This relationship has been extensively studied and bounds on convergence rate of solutions with limited number of hidden units to optimal solution (3) (e.g. [4,5,6]) have been derived. Most of the results agree on convergence rate close to $\frac{1}{\sqrt{h}}$, where h is the number of hidden units.

We studied the relation between the network size (i.e. number of hidden units) and approximation accuracy and generalization by experimental means [7]. With respect to theoretical results, we expect the approximation accuracy to improve with increasing number of hidden units. Reasonable approximation accuracy should be achieved already with small networks. In addition, high number of hidden units makes the learning task more difficult, which can influence the results.

In our experiments, we applied gradient learning on data from Proben1 repository [8]. Fig. 1 shows the results for *cancer* task. It shows the error achieved on the training

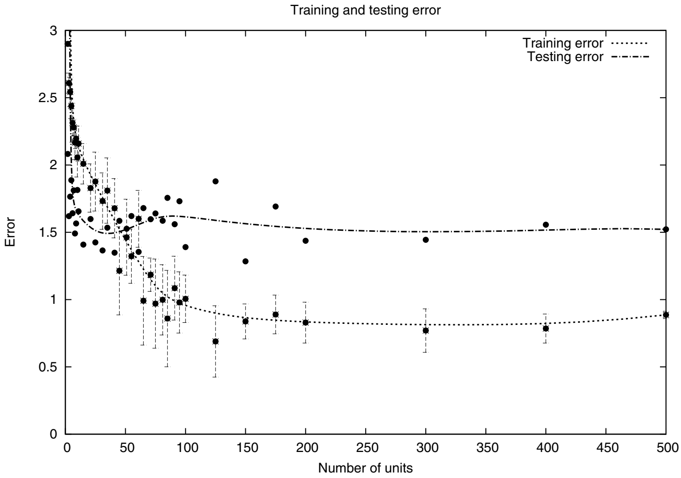


Fig. 1. Testing and training errors depending on the number of network units

set (median of 10 computations) and corresponding error on testing set. It can be seen that for small numbers of hidden units the training error increases rapidly, while for networks with more than 100 units there is no significant improvement. The situation for generalization ability represented by testing error is similar. However, the increase stops earlier, the minimal errors are achieved for networks with about 40 units.

Since the convergence is quite fast, we can suggest that small networks provide sufficiently good solutions. The theoretically estimated convergence rates justify using network of smaller complexity in real-life applications. Smaller networks have also smaller number of parameters that has to be tuned during the training process. Therefore, they are more easily trained.

4 Hybrid Learning Algorithm

In this section we propose hybrid learning algorithm creating regularization networks with small number of hidden units. It is based on a combination of evolutionary algorithms and gradient descent method.

Evolutionary algorithms have been used as a very robust and efficient search procedure in various tasks, including neural networks training [9,10]. In [11] we have shown that in the case of RBF networks, the evolutionary learning has both certain advantages and drawbacks. On one hand, evolutionary algorithms are prone to local minima stucking problem, but on the other hand, they are less computationally efficient in finding best global solution. That is the main motivation behind hybrid approaches that combine the evolutionary framework with local fine tuning operations, such as various gradient algorithms.

Our proposed hybrid algorithm (see Algorithm 4.1) works with the population of *individuals* which represent encoded parameterizations of regularization networks. By an *individual* I we understand a vector of floating-point encoded values $(c_{11}, \dots, c_{1n}\ b_1, w_1, \dots c_{h1}, \dots, c_{hn} b_h, w_h)$, where h is a number of units, n is an input dimension, $w_i \in R$ are weights of linear combination, $b_i \in R$ are widths, and $c_{ij} \in R$ are centroid positions of every unit. Such an individual I corresponds to a regularization network computing function $F_I(x) : R^n \to R$

$$F_I(x) = \sum_{i=1}^{h} w_i K_{b_i}(c_i, x)$$

The *fitness* f_I of an individual I is computed by means of an performance error over a given training set $\{(x_1, d_1), \dots (x_T, d_T)\}$, e.g.

$$f_I = 100 - 100\frac{1}{T}\sum_{t=1}^{T}\left(F_I(x_t) - d_t\right)^2 .$$

New populations are created using genetic operations of selection, mutation and crossover types. The selection operator is a fairly standard tournament selection with a small elitist rate. There are three different operators of a mutation type: a small random change of parameterization, a unit addition and a unit deletion. The first operator (cf. *mutate*) performs a small random change of individual parameters:

$$I_i = I_i + N(-1, 1),$$

The unit deletion operator (cf. *delete*) selects a substring of parameterization corresponding to randomly chosen unit in a network, and deletes a this substring. The unit addition operator (cf. *insert*) randomly generates a substring of a size of one unit in a regularization network, and inserts it into a random position on the unit borders of the original parameterization. It is clear to see that the last two operations alter the network size in a opposite manner. The crossover operator is a simple 1-point crossover on a floating point vector.

Before the fitness is evaluated, the gradient local search (cf. *gradient*) for an individual I performs n_g steps of gradient algorithm starting from the current parameterization value I. It uses the error partial derivative values to compute the gradient and then perform a small change of parameters accordingly (for details cf. [11])

$$I_i = I_i - \alpha\frac{\partial E}{\partial I_i}$$

where $\alpha > 0$ is a learning rate parameter.

The whole scheme of the hybrid algorithm is sketched in Algorithm 4.1. The starting population consists of networks with one unit only letting the evolution freedom to create bigger networks by mutation operator.

```
Create randomly an initial population $P_0$ of $M$ individuals.
$i \leftarrow 0$
For each individual $I \in P_i$:
    (a) $gradient(I)$ : perform local search by gradient descent algorithm.
    (b) evaluate fitness $f_I$.
$P_{i+1} \leftarrow$ empty set
$I_1 \leftarrow selection(P_i); I_2 \leftarrow selection(P_i)$
with probability $p_{cross}$: $(I_1, I_2) \leftarrow crossover(I_1, I_2)$
with probability $p_{mutate}$: $I_k \leftarrow mutate(I_k), k = 1, 2$
with probability $p_{delete}$: $I_k \leftarrow delete(I_k), k = 1, 2$
with probability $p_{insert}$: $I_k \leftarrow insert(I_k), k = 1, 2$
insert $I_1, I_2$ into $P_{i+1}$
if $P_{i+1}$ has less then $M$ individuals goto 5
$i \leftarrow i + 1$
goto 4 and iterate until the fitness stops increasing
```

Algorithm 4.1

5 Experiments

The hybrid algorithm was tested on tasks from Proben1 data repository [8], which contains benchmark data sets used for neural network testing. The algorithm was run 10 times and average performance was evaluated.

Table 2 lists error on training and testing set achieved by the hybrid algorithm and by Algorithm 2.1. In the terms of training errors the hybrid algorithm performed better in 4 cases, in the rest 5 cases the error is slightly higher than for regularization networks. Regarding the generalization capability, i.e. the testing error, the hybrid algorithm performed better in 6 cases. Note that the number of hidden units needed by regularization networks is much higher than the one needed by the hybrid algorithm.

Figure 2 shows the flow of error function during the run of the hybrid algorithm. Figure 3 demonstrates an evolution of the number of hidden units.

Table 2. Error values of hybrid learning algorithm compared to regularization network results

	Hybrid algorithm			RN		
Task	E_{train}	E_{test}	units	E_{train}	E_{test}	units
cancer1	1.85	1.49	8	2.28	1.75	525
cancer2	1.68	2.97	8	1.86	3.01	525
cancer3	1.77	2.78	8	2.11	2.79	525
card1	8.88	9.82	9	8.75	10.01	518
card2	7.15	13.39	15	7.55	12.53	518
card3	8.14	12.48	13	6.52	12.35	518
diabetes1	14.52	16.01	7	13.97	16.02	576
diabetes2	14.40	18.02	5	14.00	16.77	576
diabetes3	14.86	15.33	5	13.69	16.01	576

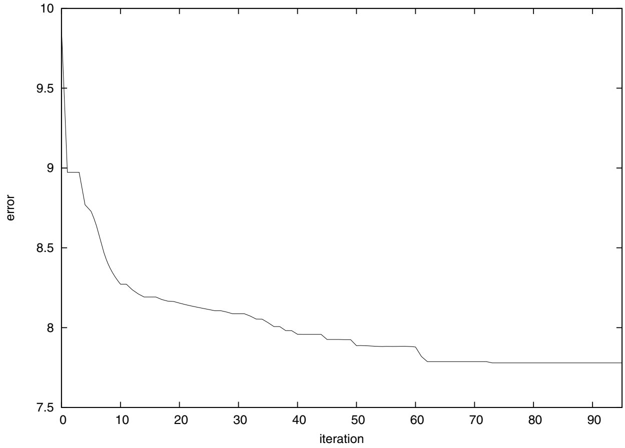


Fig. 2. Fitness function

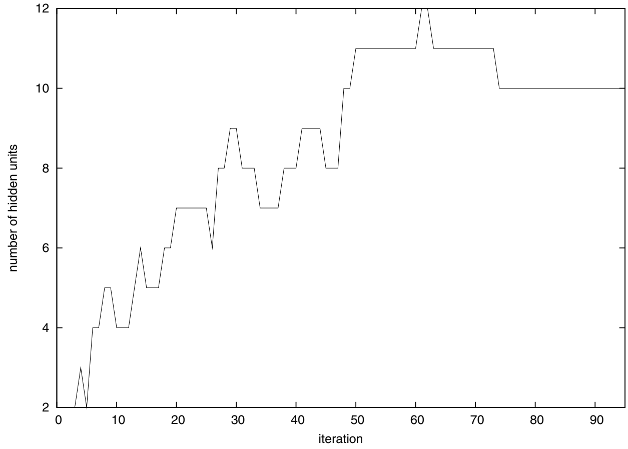


Fig. 3. The number of hidden units during evolution

6 Conclusion

Most of the learning algorithms work with networks of fixed architectures. Those optimizing also the number of hidden units can be divided into two groups – incremental and pruning. Pruning algorithm starts with large networks and tries to eliminate the irrelevant units, while incremental algorithms start with small network and add units as long as the network performance improves. The mentioned theoretical results speaks in favor of incremental algorithms. First, learning of small networks is fast since small numbers of parameters has to be optimized. Second, it is quite probable that reasonable solution will be found among smaller networks. Based on our experiments, we recommend to start with small number of hidden units and increase the network size only as long as also generalization ability improves.

The hybrid learning algorithm proposed in this work goes along the lines of previously mentioned results. The algorithm combines global optimization by widely exploring evolutionary algorithm with local exploiting search by gradient descent procedure. It has been designed as an incremental algorithm starting with minimalistic network sizes and stochastically growing the evolved networks in the course of learning. The performance of the algorithm is demonstrated on a set of experiments on the benchmark data. It can be seen that this approach achieves solutions comparable to full-size regularization networks, yet the evolved networks have typically much smaller numbers of hidden units.

Acknowledgments

This research has been fully supported by the the project KJB100300804 GA AV ČR and by the Institutional Research Plan AV0Z10300504 "Computer Science for the Information Society: Models, Algorithms, Applications".

References

1. Tikhonov, A., Arsenin, V.: Solutions of Ill-posed Problems. W.H. Winston, Washington (1977)
2. Poggio, T., Girosi, F.: A theory of networks for approximation and learning. Technical report, Cambridge, MA, USA (1989); A. I. Memo No. 1140, C.B.I.P. Paper No. 31
3. Poggio, T., Smale, S.: The mathematics of learning: Dealing with data. Notices of the AMS 50, 536–544 (2003)
4. Xu, L., Krzyżak, A., Yuille, A.: On radial basis function nets and kernel regression: statistical consistency, convergence rates, and receptive field size. Neural Netw. 7(4), 609–628 (1994)
5. Corradi, V., White, H.: Regularized neural networks: some convergence rate results. Neural Computation 7, 1225–1244 (1995)
6. Kůrková, V., Sanguineti, M.: Learning with generalization capability by kernel methods of bounded complexity. J. Complex 21(3), 350–367 (2005)
7. Vidnerová, P., Neruda, R.: Testing error estimates for regularization and radial function networks. In: Sun, F., Zhang, J., Tan, Y., Cao, J., Yu, W. (eds.) ISNN 2008, Part I. LNCS, vol. 5263, pp. 549–554. Springer, Heidelberg (2008)
8. Prechelt, L.: PROBEN1 – a set of benchmarks and benchmarking rules for neural network training algorithms. Technical Report 21/94, Universitaet Karlsruhe (November 1994)
9. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE 9(87), 1423–1447 (1999)
10. Stanley, K.O., D'Ambrosio, D., Gauci, J.: A hypercube-based indirect encoding for evolving large-scale neural networks. Artificial Life 15(2) (2009)
11. Neruda, R., Kudová, P.: Learning methods for radial basis functions networks. Future Generation Computer Systems 21, 1131–1142 (2005)

Computer Assisted Peptide Design and Optimization with Topology Preserving Neural Networks

Jörg D. Wichard[1], Sebastian Bandholtz[2], Carsten Grötzinger[2], and Ronald Kühne[1]

[1] FMP Berlin, Robert-Rössle-Str. 10, D-13125 Berlin, Germany
wichard@fmp-berlin.de
http://www.fmp-berlin.de

[2] Charité, Department of Hepatology and Gastroenterology,
Augustenburger Platz 1, D-13353 Berlin, Germany
http://www.charite.de

Abstract. We propose a non-standard neural network called TPNN which offers the direct mapping from a peptide sequence to a property of interest in order to model the quantitative structure activity relation. The peptide sequence serves as a template for the network topology. The building blocks of the network are single cells which correspond one-to-one to the amino acids of the peptide. The network training is based on gradient descent techniques, which rely on the efficient calculation of the gradient by back-propagation. The TPNN together with a GA-based exploration of the combinatorial peptide space is a new method for peptide design and optimization. We demonstrate the feasibility of this method in the drug discovery process.

1 Introduction

An important task in modern drug discovery is to understand the quantitative structure activity relation (QSAR). QSAR problems can be divided into a coding and learning part. The learning part could be solved with standard machine learning tools. Artificial neural networks are commonly used in this context as nonlinear regression models that correlate the biological activities with the physiochemical or structural properties of those chemical compounds that were tested in a specific assay.

The most important part in QSAR analysis is the identification of molecular descriptors which encode the essential properties of the compounds under investigation. Alternative approaches of the classical machine-learning-based QSAR circumvent the problem of computing and selecting a representative set of molecular descriptors. Therefore the molecules are considered as structured data - represented as graphs - wherein each atom is a node and each bond is an edge. This is the main concept of the *Molecular Graph Network* [1, 2, 3] and of the *Graph Machines* [4] which translate a chemical structure into a graph that works as topology-template for the connections of a neural network.

In this work, we follow the idea of translating the chemical structure of a compound directly into the topology of a learning machine. Our strategy is focused on peptides which are chains of amino acids. Each cell in the network corresponds one-to-one to an amino acid in the peptide. Hence the amino acid sequence of a peptide determines the

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 132–139, 2010.
© Springer-Verlag Berlin Heidelberg 2010

topology of the network. We call this architecture *Topology Preserving Neural Network* (TPNN) and we propose a learning strategy that adapts the weights of the cells with respect to the the assay data. The adapted cells are used to build models for the QSAR of new *virtual* peptides in order to optimize the desired property *in silico*. We explore the high dimensional space of all possible peptides with a genetic algorithm wherein the output of TPNN-model defines the fitness function. Only the top ranking *in silico* peptides are selected for synthesis and *in vitro* testing in the assay.

The fully connected TPNN is described in the next section and the training and regularization follows in section 3. The use of TPNN models in peptide design is reported in section 4 and first results are presented in section 5.

2 Network Representation of Peptides

Peptides are short linear polymers built from amino acids that are linked with an amide bond. The 20 proteinogenic amino acids are the most important ones and they are the building blocks of almost all proteins in nature. The string representation of the peptide S is called the *peptide sequence* and it is given by the order in which the amino acid lie in the chain. We further assume that the peptides are composed of amino acids from a pool of M different individuals called the *alphabet*. In a TPNN each amino acid from the alphabet is represented as a particular cell with individual weights that are adjusted during the network training. The internal weight of the cell is φ, the inputs from the neighboring cells are connected with the weights $\omega_{-N,\dots,N}$ and the feedback is controlled by ω_0. The weights are combined in the weight vector $\boldsymbol{\omega}$. The cells are connected to form a chain as shown in figure 1 with an one-to-one correspondence

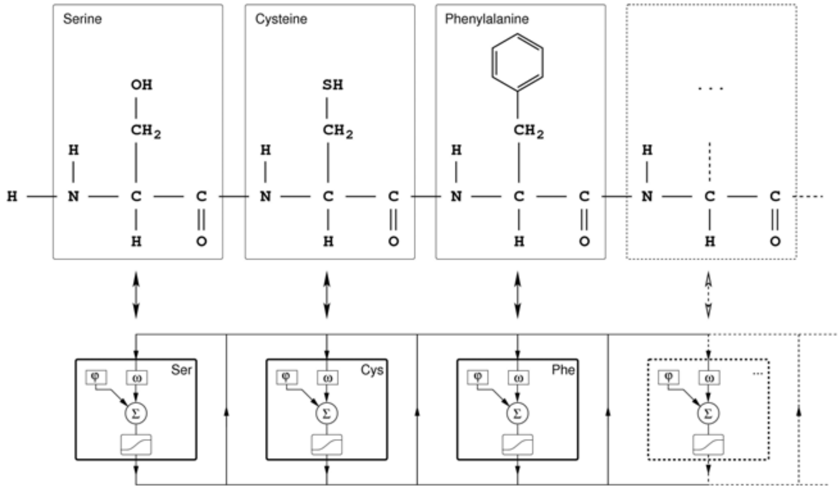


Fig. 1. This is an example of the translation process from a peptide starting with the amino acids Ser-Cys-Phe into a fully connected TPNN. Note the one-to-one correspondence between the amino acids of the peptide and the elementary cells of the TPNN.

between the amino acids in the alphabet that build the peptides and the cells in the network. The TPNN is iterated several times which governs the system dynamics. The internal states of the network are denoted in the state vector $\boldsymbol{y}$ with respect to their order. The state of the i-th TPNN cell y_i^t evolves for iterations $t = 0, \dots, T-1$ according to

$$\begin{aligned} x_i^{t+1} &= \varphi_i + \boldsymbol{y}^t \boldsymbol{\omega_i} \\ y_i^{t+1} &= \sigma(x_i^{t+1}), \end{aligned} \tag{1}$$

wherein the activation function $\sigma(x)$ is a hyperbolic tangent with an additional linear term that leads to non-vanishing derivatives of the training error

$$\sigma(x) = \tanh(x) + \lambda x \quad \text{with } \lambda \ll 1. \tag{2}$$

The number of iterations T is set to be the average length of the sequences under investigation. The final output of the network is simply the sum over all internal states y_i^T after the final iteration.

3 Training TPNN Models

The training procedure of a TPNN is a combination of stochastic gradient descend and back propagation with several improvements that make the training of the shared weights feasible. The true gradient is approximated by the gradient of the loss function which is evaluated on single training samples. The network weights are adjusted by an amount proportional to this approximate gradient. A training sample consists of two parts: The first part is the peptide sequence $\boldsymbol{S}$ that is a composition of the M possible amino acids taken from the alphabet. The second part is the measured activity that could be a continuous value or a class label, for example the classes *active*, *weak-active* or *non-active*. Let's assume that we have a collection of K training samples $\{\boldsymbol{S}_n, a_n\}_{n=1,\dots,K}$ and the weights $\boldsymbol{\omega}^j = (\varphi^j, \omega_{-N}^j, \dots, \omega_N^j)_{j=1,\dots,M}$ of the individual cells that correspond to the M different amino acids in the alphabet are organized in the weight vector $\Omega = (\boldsymbol{\omega}^1, \dots, \boldsymbol{\omega}^M)$. Let $f(\boldsymbol{S}_i, \Omega)$ denote the output of the TPNN for a given sequence $\boldsymbol{S}_i$ with respect to the network weights Ω. This output value has to be compared to the training label a_i by means of a *loss function*. The loss function measures the deviation of the TPNN output from the desired value a_i. In optimization usually a quadratic loss function is used, basically due to the simplicity of the resulting derivatives. We propose the use of an ϵ–insensitive squared loss function

$$\lambda_\epsilon(\xi) := \begin{cases} 0 & : \xi \leq \epsilon \\ (\xi - \epsilon)^2 & : \xi > \epsilon. \end{cases} \tag{3}$$

The output of the TPNN has zero loss and gradient if it lies inside the ϵ–margin of the desired output. This forces the training algorithm to focus on the training samples that are not properly explained by the current model rather than adjusting the network weights by gradient steps of already correctly learned samples. We choose an ϵ that is close to the mean of the single measurement variances.

Thus the training error $E(\Omega, \epsilon)$ is simply the loss averaged over the entire training set

$$E(\Omega, \epsilon) := \sum_{i=1}^{K} \lambda_\epsilon (a_i - f(\boldsymbol{S}_i, \Omega)). \tag{4}$$

Furthermore we need regularization in order to prevent overfitting. Weight decay is a regularization method that penalizes large weights in the network and forces the insignificant weights to converge to zero. This results in a model with a minimum number of free parameters, according to the principle of Occam's razor also known as the law of parsimony. It tells us to prefer the simplest of all equally good models. The weight decay penalty term is defined as

$$P(\Omega) = \gamma \sum_{i=1}^{N} \frac{\omega_i^2}{1 + \omega_i^2}, \tag{5}$$

where Ω denotes the weight vector of the TPNN and the regularization parameter $\gamma = 0.001$ is small. The penalty term is added to the training error and contributes to the gradient.

3.1 Stochastic Gradient Descent

Training a learning machine is put to effect by minimizing the training error as defined in equ. 4 with respect to the network weights Ω. The method of training a TPNN is based on *stochastic gradient descend*. The gradient of the entire training error from equ. 4 and the penalty term from equ. 5 is a sum of terms of the form

$$\frac{\partial}{\partial \Omega} \left[\lambda_\epsilon (a_i - f(\boldsymbol{S}_i, \Omega)) + P(\Omega) \right]. \tag{6}$$

The stochastic gradient descent performs a series of very small consecutive steps, determining each step direction from the gradient of an individual term only. After each step, the new parameter set Ω is re–inserted into the loss function before the next gradient is computed. This defines an update rule for the parameters of the form

$$\Omega_i = \Omega_{i-1} - \delta\Omega_i, \tag{7}$$

with $i = 1 \ldots K$, wherein K is the number of training samples. The update $\delta\Omega_i$ depends on the i-th training sample only and is given by

$$\delta\Omega_i = \mu \frac{\partial}{\partial \Omega} \left[\lambda_\epsilon (a_i - f(\boldsymbol{S}_i, \Omega_{i-1})) + P(\Omega_{i-1}) \right]. \tag{8}$$

We calculate the update $\delta\Omega_i$ with the standard error back-propagation technique as it is used for the common feed-forward multilayer perceptron [5].

The parameter μ controls the stepsize of the gradient descend. The initial step size is already small (around $\mu = 0.01$) and it is decreased after each training epoch with a constant factor. This is necessary to achieve a slow convergence of the weights. Note that in each training step only a few selected values of the entire weight vector Ω are adjusted, namely the ones that correspond to amino acids that appear in the sequence of the training sample.

3.2 Building TPNN-Ensembles

A common way to improve the performance of neural networks in regression or classification tasks is ensemble building [6]. It is well known, that neural network ensembles perform better in terms of generalization than single models would do [7, 8]. An ensemble of TPNNs consists of several single TPNN models that are trained on randomly chosen subsets of the training data and the training starts with random weight initializations. This ensures the diversity of the resulting models which is the key issue in ensemble building. To compute the output of the ensemble for one input sequence, the output variables of all TPNNs belonging to the ensemble are averaged. We build ensembles with 20-30 individual trained TPNN models.

4 Computer Assisted Peptide Design with TPNN Models

The main objective in building a TPNN model is to recover the fundamental characteristics of the structure activity relation. Therefore a start population of peptides is selected. The sequence strings of the peptides together with the measurements from the biological assay deliver the data for TPNN training as described in section 3. Training the network means adapting the weights of the cells that correspond to the amino acids of the peptides in the training set. The adapted cells work as building blocks of new *virtual* peptides which are generated by rearranging the order of the cells and calculating the output of the network according to equ. 1. The resulting TPNN-model defines the fitness function in a genetic algorithm (GA) that is generating new suggestions for peptide synthesis based on the learned structure activity relation. The reason for the GA approach is the huge dimension of the ”peptide space” that we have to explore. In this

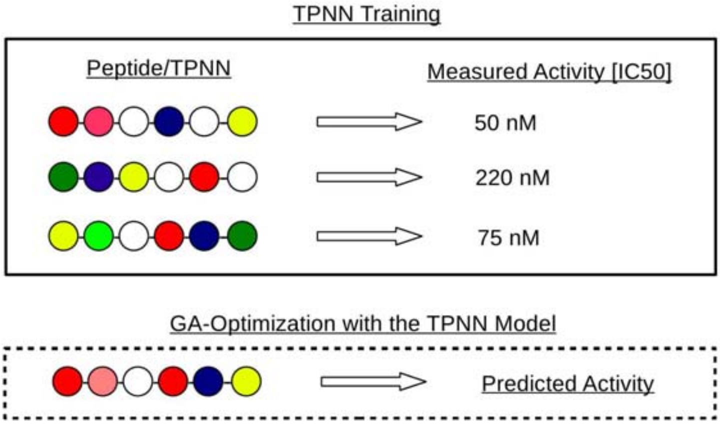


Fig. 2. The example shows how the TPNN model works in computer assisted peptide design: The model learns the properties of the peptides in the training set and the adopted cells build *virtual* peptides that are evaluated in the GA-optimization

study we investigate 9-mer peptides from an alphabet that consists of the 20 natural amino acids and 15 non-natural D-amino-acids. We include D-amino acids in order to increase the metabolic stability of the peptides. This leads to combinatorial library of $35^9 \approx 7.8 \times 10^{13}$ possible peptides.

The GA represents each chromosome as amino acid sequence following the building block theory introduced by Holland [9] and Goldberg [10]. We perform mutation and 2-point crossover on the sequence level. The start population consists of 2000 randomly generated 9-mer peptides and evolves over 5000 generations. We prefer to use *elitist selection* by keeping the best performing individuals of the population unchanged. The new GA-based peptide suggestions are synthesized and tested in the biological assay and the results are included in the next TPNN training cycle. This process is repeated several times and improves the peptides in each round with respect to the desired properties.

5 Results

The first target to test our approach was a human G-protein-coupled receptor (GPCR) from the rhodopsin family. G-protein-coupled receptors are a protein family of transmembrane receptors that modulate several vital physiological events and comprise one of the largest families in the human genome with more than 800 identified sequences [11]. The involvement in many biological processes has the consequence that GPCRs play a key role in several pathological conditions, which has led to GPCRs being the target of up to 40% of today's marketed drugs [12, 13]. Nevertheless very little is known about the structure and structure-function relationship of this important target family because up to now there are only three mammalian GPCR crystal structures published[1]. The lack of structural knowledge is the reason why the common structure based modelling techniques cannot be used without difficulties. The advantage of the TPNN approach is that it makes no assumption about the structural features of the drug target. This method works without any explicit structural information.

The goal of our approach was the development of a 9-mer peptide with high activity and metabolic stability. The activity of the peptides was measured in a functional Cellux-assay on Ca^{2+} mobilization using stably transfected HEK293tet cells expressing the human GPCR. We applied eleven different concentrations of the peptides to obtain concentration response curves for EC_{50} calculations. All EC_{50} values are results of 3 experiments made in duplicates. The metabolic stability was measured via reverse phase HPLC with a ZORBAX Eclipse XDB-C18, $4,6 \times 150$ mm, 5 μm column. For that the peptides had been incubated in 25% human serum at 37°C. Samples were analyzed at different time points to determine the half life (in minutes) of the peptides in the human serum.

We started with a random population of 29 peptides followed by three optimization cycles with 30-50 peptides each. The results with respect to the EC_{50} values of the activity and the metabolic stability of the compounds are shown in figure 3.

[1] The GPCR crystal structures so far are the inactive conformation of bovine-rhodopsin [14, 15], the human β_2-adrenergic receptor [16] and the human A2A adenosine receptor [17].

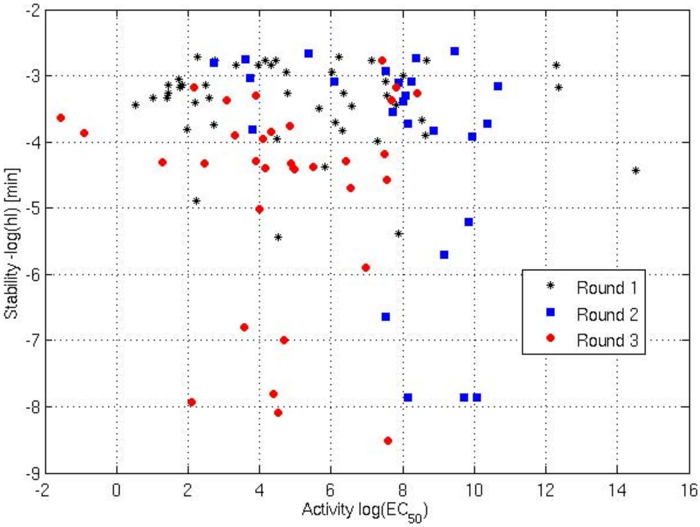


Fig. 3. This figure shows the metabolic stability (negative logarithm of the half life in minutes) versus the log(EC_{50}) values of the activity for the first three optimization rounds. The trend points to the lower left corner of the diagram, i.e. peptides that combine activity with high metabolic stability.

6 Conclusion

We introduced a new topology preserving cellular neural network that operates on single cells as building blocks with a one-to-one correspondence to the amino acids of the peptide. The TPNN mimics the chain structure of a peptide and translates a chemical structure directly into the topology of a learning machine. This overcomes the obstacle of designing and computing molecular descriptors for the QSAR. Furthermore the TPNN does not rely on the availability of structural information about the drug target.

The concept of TPNN together with the GA-based exploration of the combinatorial peptide space is the core concept of a novel peptide optimization process in drug discovery. The feasibility of the design approach was demonstrated for the construction and optimization of peptidic GPCR ligands in an iterative process of 3 design cycles of computer assisted optimization with respect to the activity and the metabolic stability. Synthesis and experimental fitness determination of less than 160 different compounds from a virtual combinatorial library of more than 7.8×10^{13} peptides were necessary to achieve this goal.

This is work in process. We plan to run further optimization cycles in order to improve activity and metabolic stability of the compounds. We cannot disclose the target and the peptide sequences before the investigation is finished and the patent situation is clarified.

Acknowledgment

The authors would like to thank the members of the Molecular Modelling Group at FMP Berlin and the members of the AG Tumortargeting at Charité Berlin.

References

1. Ogorzałek, M., Merkwirth, C., Wichard, J.: Pattern recognition using finite-iteration cellular systems. In: Proceedings of the 9th International Workshop on Cellular Neural Networks and Their Applications, pp. 57–60 (2005)
2. Merkwirth, C., Lengauer, T.: Automatic generation of complementary descriptors with molecular graph networks. Journal of Chemical Information and Modeling 45(5), 1159–1168 (2005)
3. Merkwirth, C., Ogorzałek, M.: Applying CNN to cheminformatics. In: Proceedings of the ISCAS, pp. 2918–2921 (2007)
4. Goulon, A., Picot, T., Duprat, A., Dreyfus, G.: Predicting activities without computing descriptors: Graph machines for QSAR. SAR and QSAR in Environmental Research 18, 141–153 (2007)
5. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature 323, 533–536 (1986)
6. Hansen, L., Salamon, P.: Neural network ensembles. IEEE Trans. on Pattern Analysis and Machine Intelligence 12(10), 993–1001 (1990)
7. Geman, S., Bienenstock, E., Doursat, R.: Neural networks and the bias/variance dilemma. Neural Computation 4, 1–58 (1992)
8. Perrone, M.P., Cooper, L.N.: When networks disagree: Ensemble methods for hybrid neural networks. In: Mammone, R.J. (ed.) Neural Networks for Speech and Image Processing, pp. 126–142. Chapman-Hall, Boca Raton (1993)
9. Holland, J.H.: Adaptation in natural and artificial systems. MIT Press, Cambridge (1975)
10. Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning. Kluwer Academic Publishers, Boston (1989)
11. Fredriksson, R., Lagerström, M., Lundin, L., Schiöth, H.: The G-protein-coupled receptors in the human genome form five main families. Phylogenetic analysis, paralogon groups, and fingerprints. Mol. Pharmacol. 63, 1256–1272 (2003)
12. Hopkins, A., Groom, C.: The druggable genome. Nature Reviews Drug Discovery 1(9), 727–730 (2002)
13. Filmore, D.: It's a GPCR world. Modern Drug Discovery 7(11), 24–28 (2004)
14. Palczewski, K., Kumasaka, T., Hori, T., Behnke, C.A., Motoshima, H., Fox, B., Trong, I.L., Teller, D., Okada, T., Stenkamp, R., Yamamoto, M., Miyano, M.: Crystal structure of rhodopsin: A G protein-coupled receptor. Science 289(5480), 739–745 (2000)
15. Teller, D., Okada, T., Behnke, C., Palczewski, K., Stenkamp, R.: Advances in determination of a high-resolution three-dimensional structure of rhodopsin, a model of G-protein-coupled receptors. Biochemistry 40, 7761–7772 (2001)
16. Rasmussen, S., Choi, H., Rosenbaum, D., Kobilka, T., Thian, F., Edwards, P., Burghammer, M., Ratnala, V., Sanishvili, R., Fischetti, R., Schertler, G., Weis, W., Kobilka, B.: Crystal structure of the human β-2 adrenergic G-protein-coupled receptor. Nature 450, 383–387 (2007)
17. Jaakola, V., Griffith, M., Hanson, M., Cherezov, V., Chien, E., Lane, J., IJzerman, A., Stevens, R.: The 2.6 Angstrom Crystal Structure of a Human A2A Adenosine Receptor Bound to an Antagonist. Science 322(5905), 1211–1217 (2008)

Part II

Evolutionary Algorithms and Their Applications

Evolutionary Designing of Logic-Type Fuzzy Systems*

Marcin Gabryel[1,2] and Leszek Rutkowski[3,1]

[1] Department of Computer Engineering, Częstochowa University of Technology
Al. Armii Krajowej 36, 42-200 Częstochowa, Poland

[2] The Professor Kotarbinski Olsztyn Academy of Computer Science and Management
ul. Artyleryjska 3c, 10-165 Olsztyn, Poland

[3] Academy of Management (SWSPiZ), Institute of Information Technology,
ul. Sienkiewicza 9, 90-113 Lodz, Poland

marcin.gabryel@kik.pcz.pl, lrutko@kik.pcz.czest.pl

Abstract. In this paper we present a method for designing the logic-type fuzzy system. In this kind of fuzzy systems antecedents and consequences, in the individual rules, are connected by a fuzzy implication. In our method, the whole system is designed by an evolutionary algorithm, including learning of parameters of membership functions and selection of an appopriate fuzzy implication and triangular norms. The results of simulations illustrate efficiency of our method.

1 Introduction

In the literature various architectures of neuro-fuzzy systems have been studied [8][9][11]. The Takagi-Sugeno scheme involves a functional dependence between consequents and inputs. In the Mamdani approach antecedents and consequences, in the individual rules, are connected by a t-norm. In the case of the logical approach antecedents and consequences are connected by a fuzzy implication, and the most popular are S, R and Q fuzzy implications [11].

In recent works on the evolutionary design of fuzzy systems [1][2][4] Mamdani-type fuzzy systems are studied. In this paper we show the result of research on the evolutionary learning of logic-type fuzzy systems. Besides the learning method applied to find parameters of membership functions, we propose a new approach to designing structure of the fuzzy system. In particular, by an evolutionary method we select a fuzzy implication and t-norms describing the fuzzy system. In view of a great number of available fuzzy implications it is possible to build various systems. A challenging problem is to choose an appropriate system for the specific task. Fortunately, we can successfully use the evolutionary strategy to optimize the structure of the whole system, that is to choose the appropriate

* This work was partly supported by the Foundation for Polish Science (TEAM project for the years 2010-2014) and the Polish Ministry of Science and Higher Education (Polish-Singapore research project for the years 2008-2010, research project for the years 2008-2011).

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 143–148, 2010.
© Springer-Verlag Berlin Heidelberg 2010

fuzzy implication, t-norms and parameters of the system. Our method completely relieves the designer of the monotonous construction of the system.

Our paper is divided into three main sections. In the next section we will give a short description of the logic-type fuzzy system. Next, we will present details of evolutionary designing method. In the last section the results of simulations are shown.

2 Logic-Type Fuzzy System

In this paper, we consider a multi input, single output fuzzy system mapping $\mathbf{X} \to \mathbf{Y}$ where $\mathbf{X} \subset \mathbf{R}^n$ and $\mathbf{Y} \subset \mathbf{R}$. The fuzzy rule base consists of a collection of N fuzzy IF-THEN rules in the form:

$$R^{(r)}: \text{ IF } x_1 \text{ is } A_1^r \text{ AND} \ldots \text{AND } x_n \text{ is } A_n^r \text{ THEN } y \text{ is } B^r, \tag{1}$$

where $\mathbf{x} = [x_1, \ldots, x_n]$ are input variables, n - number of inputs, y - output value, fuzzy sets $A_1^r, A_2^r, \ldots, A_n^r$ and B^r are characterized by membership functions $\mu_{A_i^r}(x_i)$ and $\mu_{B^r}(y)$, respectively, $r = 1, ..., N$, $i = 1, ..., n$. We use the most common singleton fuzzyfier for mapping crisp values of inputs variables into fuzzy sets [10]. The defuzzyfication process was made by the COA (center of the area method)

$$y = \frac{\sum_{r=1}^{N} \overline{y}^r \cdot \mu_{B'}(\overline{y}^r)}{\sum_{r=1}^{N} \mu_{B'}(\overline{y}^r)} \tag{2}$$

where $\overline{y}^r$ is a centre of gravity (centroid) of fuzzy set B^k, and

$$B' = \bigcap_{k=1}^{N} B^k, \quad \mu_{B'}(y) = \mathop{T}_{k=1}^{N} \{\mu_{B^k}(y)\} \tag{3}$$

Antecedents and consequences in the individual rules are connected by a fuzzy implication denoted by $I_{fuzzy}(\cdot)$. The aggregation of sets $B^k \subseteq \mathbf{Y}$ is made using a t-norm

$$\mu_{B'}(\overline{y}^r) = \mathop{T}_{k=1}^{N} \left\{\mu_{\overline{B}^k}(\overline{y}^r)\right\} = \mathop{T}_{k=1}^{N} \left\{I_{fuzzy}\left(\mu_{A^k}(\overline{\mathbf{x}}), \mu_{B^k}(\overline{y}^r)\right)\right\}, \tag{4}$$

$\overline{\mathbf{x}} = [\overline{x}_1, \ldots, \overline{x}_n]$ is the input signal. Antecedents are connected by a t-norm

$$\mu_{A^k}(\overline{\mathbf{x}}) = \mathop{T}_{k=1}^{n} \left\{\mu_{A_i^k}(\overline{x}_i)\right\}. \tag{5}$$

Combining formulas (2)-(5) we obtain the following description

$$y = \frac{\sum_{r=1}^{N} \overline{y}^r \cdot \mathop{T}_{k=1}^{N} \left\{I_{fuzzy}\left(\mathop{T}_{i=1}^{n} \left(\mu_{A_i^k}(\overline{x}_i)\right), \mu_{B^k}(\overline{y}^r)\right)\right\}}{\sum_{r=1}^{N} \mathop{T}_{k=1}^{N} \left\{I_{fuzzy}\left(\mathop{T}_{i=1}^{n} \left(\mu_{A_i^k}(\overline{x}_i)\right), \mu_{B^k}(\overline{y}^r)\right)\right\}}, \tag{6}$$

In the sequel the following implications will be used [10]:

1) Kleene-Dienes (binary)

$$I(a,b) = \max\left\{1-a, b\right\}, \tag{7}$$

2) Lukaszewicz

$$I(a,b) = \max\left\{1, 1-a+b\right\}, \tag{8}$$

3) Reichenbach

$$I(a,b) = 1-a+a\cdot b, \tag{9}$$

4) Fodor

$$I(a,b) = \begin{cases} 1 & a \leq b \\ \max\{1-a, b\} & a > b \end{cases}, \tag{10}$$

5) Goguen

$$I(a,b) = \begin{cases} 1 & a = 0 \\ \max\{1, \frac{b}{a}\} & a > 0 \end{cases}, \tag{11}$$

6) Gödel

$$I(a,b) = \begin{cases} 1 & a \leq b \\ b & a > b \end{cases}, \tag{12}$$

6) Zadeh

$$I(a,b) = \max\left\{\min(a,b), 1-a\right\}, \tag{13}$$

7) Rescher

$$I(a,b) = \begin{cases} 1 & a \leq b \\ 0 & a > b \end{cases}, \tag{14}$$

8) Yager

$$I(a,b) = b^{a}, \tag{15}$$

9) Willmott

$$I(a,b) = \max\left\{\max(1-a,b), \max(a, 1-b, \min(1-a,b))\right\}, \tag{16}$$

10) Dubois-Prade

$$I(a,b) = \begin{cases} 1-a & b = 0 \\ b & a = 1 \\ 1 & \text{other case} \end{cases}. \tag{17}$$

We chose as membership functions $\mu_{A_i^k}(x_i)$ and $\mu_{B^k}(y)$ the Gaussian functions $\mu_{A_i^k}(x_i) = \exp\left[-(x_i - \overline{x}_i^k)^2/{\sigma_i^k}^2\right]$, $\mu_{B_i^k}(y) = \exp\left[-(y-\overline{y}^k)^2/{\sigma^k}^2\right]$.

The system described by formula (6) has been trained using the idea of the backpropagation method [11]. In the next section we will describe an evolutionary algorithm to train and design such system.

3 Evolutionary Learning and Designing of Logic-Type Fuzzy Systems

3.1 Parameters Learning

Let $\overline{\mathbf{x}} \in \mathbf{X} \subset \mathbf{R}^n$. Let $y(t) \in \mathbf{Y} \subset \mathbf{R}$, $d(t) \in \mathbf{Y} \subset \mathbf{R}$, $t = 1, ..., K$, be a sequence of inputs and desirable output signals, respectively. Based on the learning sequence $((\mathbf{x}(1), d(1)),\ (\mathbf{x}(2), d(2)), ..., (\mathbf{x}(K), d(K)))$ we wish to determine all parameters $\overline{x}_i^k$, σ_i^k, $\overline{y}^k$, σ^k, $i = 1, ..., n$, $k = 1, ..., N$, such that $e(t) = \frac{1}{2}\left[y(t) - d(t)\right]^2$ is minimized, where y is given by formula (6).

We will solve the above problem by using an evolutionary strategy (μ, λ) [3]. This method consists in selection of system parameters describing shapes of membership functions. Parameters $\overline{x}_i^k$, σ_i^k, $\overline{y}^k$, σ^k are encoded in a chromosome as the real-value numbers. After using genetic operators and determining values of fitness functions, which reflect the effectiveness of the coded systems, the evolutionary algorithm selects the best individuals to the next generation.

The learning metod of fuzzy systems needs to initiate the primary population of individuals. It usually consists in random drawing of values of respective genes. In this paper a method that allows to initiate both the chromosomes coding system parameters and a chromosome describing values of the range of mutation has been used. The method accelerates the process of optimization. In some cases the method makes possibile to complete this process in a reasonable time what is not assured by previous methods of evolutionary alorithms initialization [3][7]. The learning algorithm and the algorithm of initialization of primary population are presented in detail in our previous works [5][6].

In the fuzzy system given by formula (6) membership functions are described by parameters $(\overline{x}_i^k, \sigma_i^k)$ and $(\overline{y}^k, \sigma^k)$, respectively. Thus each of the rules will be encoded in a piece of the chromosome $\mathbf{X}_j$ denoted by $\mathbf{X}_{j,k} = \left(\overline{x}_1^k, \sigma_1^k, \overline{x}_2^k, \sigma_2^k, \ldots, \overline{x}_n^k, \sigma_n^k, \overline{y}^k, \sigma^k\right)$, where $k = 1, \ldots, N$ and j is a number of the chromosome. The complete rule base is represented by chromosome $\mathbf{X}_j = (\mathbf{X}_{j,1}, \mathbf{X}_{j,2}, \ldots, \mathbf{X}_{j,N})$. Details are presented in [4].

3.2 Designing the Structure of the Logic-Type Fuzzy System

The encoding method outlined in the previous subsection will be extended by using an additional chromosome which encodes information about t-norms and fuzzy implications used in the logic type fuzzy system. It may be noted that due to the different combinations of these functions, there can occur quite a number of different systems. Ten implications (7)-(17) and two t-norms (product and minimum) create 30 different combinations of systems. For a designer, who chooses the logical reasoning model, it is time consuming to test the performance of all combinations. Therefore, we propose the appropriate evolutionary algorithm, which is able to select the most appropriate structure of the system and also to choose the parameters of membership functions.

Schematic process of encoding is shown in Figure 1. New chromosome, denoted by $\mathbf{k}_j$, has 3 genes, which take integer values. The first gen is responsible

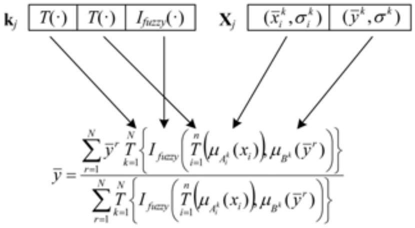

$\mathbf{k}_j$
$T(\cdot)$
$T(\cdot)$
$I_{fuzzy}(\cdot)$
$\mathbf{X}_j$
$(\bar{x}_i^k, \sigma_i^k)$
$(\bar{y}^k, \sigma^k)$
$\bar{y} = \frac{\sum_{r=1}^{N} \bar{y}^r \overset{N}{\underset{k=1}{T}} \left\{ I_{fuzzy}\left(\overset{n}{\underset{i=1}{T}} \left(\mu_{A_i^k}(x_i) \right), \mu_{B^k}(\bar{y}^r) \right) \right\}}{\sum_{r=1}^{N} \overset{N}{\underset{k=1}{T}} \left\{ I_{fuzzy}\left(\overset{n}{\underset{i=1}{T}} \left(\mu_{A_i^k}(x_i) \right), \mu_{B^k}(\bar{y}^r) \right) \right\}}$

Fig. 1. Schematic process of encoding

for encoding t-norms combining antecedents (5). The second gene encodes a t-norm which is used to aggregate fuzzy sets B^k in formula (3). The last gene is responsible for encoding the type of fuzzy implication $I_{fuzzy}(\cdot)$.

Each type of function is encoded using a numeric value. The product t-norm is marked by 0 wheras the minimum t-norm by 1. In the case of fuzzy implications (7)–(17), the implications are sequentially numbered: Kleene-Dienes is marked by 0, Lukasiewicz by 1, etc. and Dubois-Prade by 10. During the evolution on the new chromosome $\mathbf{k}_j$ is performed only one genetic operation – a mutation. The mutation occurs with some probability p_m and results in an increase of the gene value by one. In case when the gene value is equal to the number of the last encoded function (t-norm or fuzzy implication), then it takes the value equal to zero.

4 Symulation Results

Our evolutionary method for learning and designing logic-type fuzzy systems was simulated on the Glass problem, Monk1 problem and Wine recognition problem [12]. We assume the following parameters for simulations: $\mu = 100$, $\lambda = 500$, and maximum number of generations is 500. The experimental results are shown in Table 1. Subsequent colums contain information about: problem name, selected fuzzy implication, selected t-norms for conneccting antecedents in formula (5) and aggregation of sets B^k in formula (3) and effectiveness of the classification (percentage of correctly classified samples) obtained in the learning and testing processes. We obtained the full effectiveness of the Monk1 classification and good results for the glass and wine classification.

Table 1. The experimental results

problem	implication	t-norm	t-norm	training	testing
Glass	Goguen	product	product	100.0%	85.9%
Monk1	Goguen	min	min	100.0%	100.0%
Wine	Gödel	product	min	100.0%	94.23%

5 Final Remarks

In logic-type systems, the antecedents and consequences of rules are connected by fuzzy implications. Combining fuzzy implications (7)–(17) with appopriate t-norms, performing operations (3) and (5), we obtain various fuzzy systems. In the paper the evolutionary method of choosing the best fuzzy system for a specific solved problem was presented. Simulation results confirm the effectiveness of our method. By making use of the evolutionary aproach we designed the structure of a fuzzy system and learned its parameters.

References

1. Cordon, O., Herrera, F., Hoffman, F., Magdalena, L.: Genetic Fuzzy System. Evolutionary Tunning and Learning of Fuzzy Knowledge Bases. World Scientific, Singapore (2000)
2. Cordon, O., Gomide, F., Herrera, F., Hoffmann, F., Magdalena, L.: Ten years of genetic fuzzy systems: current framework and new trends. Fuzzy sets and systems 141, 5–31 (2004)
3. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing. Springer, Heidelberg (2003)
4. Gabryel, M., Cpalka, K., Rutkowski, L.: Evolutionary strategies for learning of neuro-fuzzy systems. In: I Workshop on Genetic Fuzzy Systems, Genewa, pp. 119–123 (2005)
5. Gabryel, M., Rutkowski, L.: Evolutionary Learning of Mamdani-type Neuro-Fuzzy Systems. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 354–359. Springer, Heidelberg (2006)
6. Korytkowski, M., Gabryel, M., Rutkowski, L., Drozda, S.: Evolutionary Methods to Create Interpretable Modular System. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 405–413. Springer, Heidelberg (2008)
7. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs, 3rd edn. Springer, Heidelberg (1996)
8. Rutkowska, D., Nowicki, R.: Implication-Based Neuro-Fuzzy Architectures. Intenrational Journal of Applied Mathematics and Computer Science 10(4) (2000)
9. Rutkowska, D.: Neuro Fuzzy Architectures and Hybrid Learning. Springer, Heidelberg (2002)
10. Rutkowski, L.: Computational Inteligence. Methods and Techniques. Springer, Heidelberg (2008)
11. Rutkowski, L.: Flexible Neuro Fuzzy Systems. Kluwer Academic Publishers, Dordrecht (2004)
12. Mertz, C.J., Murphy, P.M.: UCI respository of machine learning databases, http://www.ics.uci.edu/pub/machine-learning-databases

Combining Evolutionary and Sequential Search Strategies for Unsupervised Feature Selection

Artur Klepaczko and Andrzej Materka

Technical University of Lodz, Institute of Electronics
ul. Wolczanska 211/215, 90-924 Lodz, Lodz
{aklepaczko,amaterka}@p.lodz.pl

Abstract. The research presented in this paper aimed at development of a robust feature space exploration technique for unsupervised selection of its subspace for feature vectors classification. Experiments with synthetic and textured image data sets show that current sequential and evolutionary strategies are inefficient in the cases of large feature vector dimensions (reaching the order of 10^2) and multiple-class problems. Thus, the proposed approach utilizes the concept of hybrid genetic algorithm and adopts it for specific requirements of unsupervised learning.

Keywords: Unsupervised feature selection, hybrid genetic algorithm, texture analysis.

1 Introduction

Literature describes two general approaches to the feature selection problem [1]. In the *filter* strategy an attribute saliency is estimated individually according to some evaluation measure. The main advantage of these methods is their low computational complexity as they avoid time-consuming feature space exploration. However, it is often observed that discriminative power of a single feature emerges only when it is accompanied by some other attributes.

The second option is the *wrapper* approach, in which performance of a classifier is iteratively evaluated across different feature subsets. This in turn suggests how a certain combination of attributes is useful to discriminate classes. Efficiency of wrapper-type selection strongly depends the strategy of constructing subsequent feature subsets. Especially for high-dimensional patterns, testing all possible feature combinations is intractable. Hence, a search algorithm should intelligently decide which feature subspaces should be examined to ensure convergence near optimum without performing a complete browse.

This paper focuses on wrapper-type feature selection in *unsupervised* learning. The reasons that motivate research in this area are twofold. First of all, existing search strategies which proved effective in the supervised feature selection often occur inefficient in the unsupervised mode. Secondly, some feature space exploration techniques demand target dimensionality to be determined in advance. However, in the case of unsupervised learning ambiguity concerning number of clusters involves uncertainty about minimal number of features required to properly discriminate them.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 149–156, 2010.
© Springer-Verlag Berlin Heidelberg 2010

2 Unsupervised Feature Selection

2.1 Problem Formulation

Let N denote dimension of data vectors whose components correspond to features $\xi_j, j = 1, \ldots, N$. Feature selection constructs a subset $\hat{\boldsymbol{\Xi}}$ of the most significant attributes from $\boldsymbol{\Omega} = \{\xi_1, \ldots, \xi_N\}$ of all N features. Formally:

$$\hat{\boldsymbol{\Xi}} = \underset{\boldsymbol{\Xi}_l \subset \boldsymbol{\Omega}}{\arg\max} J(\boldsymbol{\Xi}_l), \tag{1}$$

where J denotes a criterion function used to evaluate feature subset significance, $\boldsymbol{\Xi}_l \in \mathbf{M}_{\boldsymbol{\Omega}}$ — the collection of all subsets of $\boldsymbol{\Omega}$, and $l = 1, \ldots, |\mathbf{M}_{\boldsymbol{\Omega}}|$.

In the case of unsupervised learning — due to the lack of class labels in the training data set — the common choice for J are various clustering quality measures, like DB index [2] or silhouette value [3]. Such inference assumes that high quality clusters are feasible only in a subspace of relevant attributes.

Apart from a chosen quality index, a decisive element of the wrapper approach is a search strategy that constructs feature subsets $\boldsymbol{\Xi}_l$ — candidates for selection. There are three general groups of feature space exploration schemes: complete, sequential and evolutionary [4]. In the following we concentrate on evolutionary techniques which appear particulary useful in unsupervised mode. This is mainly due to their ability to escape from local optima of the objective function which are very common for clustering evaluation measures [5,3].

2.2 Evolutionary Search Strategies

The paradigm of evolutionary optimization algorithms assumes that candidate solutions are represented by population of chromosomes (or agents). In the feature selection problem, agents can be viewed as binary vectors composed of genes corresponding to all attributes in $\boldsymbol{\Omega}$. If a bit (gene) is set, then its corresponding feature is treated as significant and vice versa. Each chromosome is associated with its fitness value proportional to classification quality obtained in attribute subspace selected by unit genes. In order to browse through feature space, evolutionary techniques utilize operations which imitate natural genetic processes of living organisms — parent-chromosomes selection, cross-over, mutation and replacement (see Fig. 1 for the general outline of the Simple Genetic Algorithm — SGA) — refer e.g. to [6].

According to earlier remarks, unsupervised learning mode raises additional challenge for the feature selection problem. In majority of real-life applications the number of clusters is unknown and needs to be determined together with features significance. In [5] and [7] this issue is addressed by enlarging a chromosome with extra genes responsible for encoding number of clusters K. The length of this second bit string equals $K_{\max} - K_{\min}$, where $K_{\min}$ and $K_{\max}$ denote low and top bound settled a priori for K. Each unit gene in this part of an agent increases the number of clusters by one in relation to the low bound.

Moreover, in many approaches the task of unsupervised feature selection is considered as a multi-optimization problem. Apart from clustering quality maximization, they try to minimize feature space dimensionality and cluster complexity. However, multi-criterion optimization requires a more sophisticated search strategy than SGA. Morita et al. [7] adopt the Nondominated Sorting Genetic Algorithm (NSGA) [8], while [5] introduces original approach of feature space exploration — the Evolutionary Local Selection Algorithm (ELSA).

3 Hybrid Genetic Algorithms

An important drawback of evolutionary strategies is low convergence rate near the best solution. The same reasons which allow a genetic algorithm escape from a local optimum can cause the algorithm to abandon the global one. A hybrid approach, that combines sequential and random search schemes, provide a trade-off between the need of fast convergence and resistance to local optima.

The concept of Hybrid Genetic Algorithm (HGA) in the context of supervised learning is described in [9]. It introduces local sequential search operator invoked between mutation and replacement. The aim of this operator is to explore solution space in the neighborhood of the offspring obtained after mutation, and substitute it with a higher fitness chromosome. Below, we first describe the approach proposed in [9]. It will be referred to as fixed-dimensionality HGA (F-HGA), as it demands cardinality d of target feature subspace to be fixed a priori. Then, we introduce our variant of HGA, which relaxes the requirements for d — only its top bound $d_{\max}$ needs to be assumed. Thus, it will be called constrained-dimensionality HGA (C-HGA).

3.1 Fixed-Dimensionality HGA

As mentioned earlier, the mutated offspring-chromosome is passed to the local improvement operator. In F-HGA this stage consists of two basic operations, the so-called ripple removal (*ripple_rem*) and ripple addition (*ripple_add*), both dependent on an integer parameter r. Before local improvement starts, all features

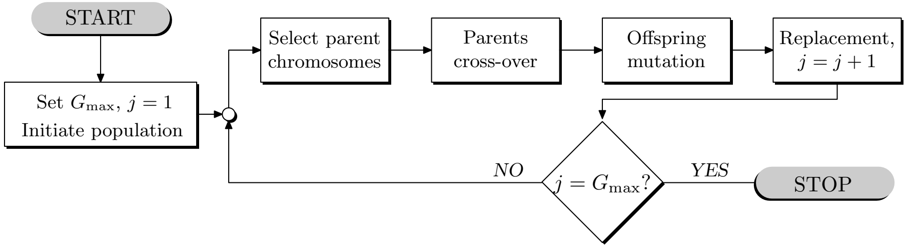


Fig. 1. Simple Genetic Algorithm with steady-state replacement method. $G_{\max}$ denotes maximal number of generations.

```
$\boldsymbol{\Xi}_0 = \boldsymbol{\Xi}; \boldsymbol{\Delta}_0 = \boldsymbol{\Delta}$
for $t = 1$ to $r$        // additions loop
    $\hat{\delta} = \arg\max_{\delta \in \boldsymbol{\Delta}_{t-1}} J(\boldsymbol{\Xi}_{t-1} \cup \{\delta\})$;
    $\boldsymbol{\Xi}_t = \boldsymbol{\Xi}_{t-1} \cup \{\hat{\delta}\};\ \boldsymbol{\Delta}_t = \boldsymbol{\Delta}_{t-1} \setminus \{\hat{\delta}\}$;
$\boldsymbol{\Xi}_0 = \boldsymbol{\Xi}_t; \boldsymbol{\Delta}_0 = \boldsymbol{\Delta}_t$;
for $t = 1$ to $r - 1$        // removals loop
    $\hat{\xi} = \arg\max_{\xi \in \boldsymbol{\Xi}_{t-1}} J(\boldsymbol{\Xi}_{t-1} \setminus \{\xi\})$;
    $\boldsymbol{\Xi}_t = \boldsymbol{\Xi}_{t-1} \setminus \{\hat{\xi}\};\ \boldsymbol{\Delta}_t = \boldsymbol{\Delta}_{t-1} \cup \{\hat{\xi}\}$;
$\boldsymbol{\Xi} = \boldsymbol{\Xi}_t; \boldsymbol{\Delta} = \boldsymbol{\Delta}_t$;
```

(a)

```
if $n_c = d$ then
    ripple_rem(r);
    ripple_add(r);
else if $n_c < d$ then
    for $i = 1$ to $(d - n_c)$
        ripple_add(r);
else if $n_c > d$ then
    for $i = 1$ to $(n_c - d)$
        ripple_rem(r);
```

(b)

Fig. 2. Pseudo-code of (a) the *ripple_add* operation and (b) the local improvement operator in F-HGA

whose corresponding genes in the offspring equal to one, are put into a significant feature subset $\boldsymbol{\Xi}$. All the remaining attributes form insignificant subset $\boldsymbol{\Delta}$. Then, the *ripple_add* operation proceeds as depicted in Fig. 2a.

The *ripple_rem* procedure consists of similar loops as *ripple_add*, but in reverse order. It starts with r consecutive removals and is followed by $r - 1$ additions. Basing on these two generic procedures the local improvement operation depends on the relation between number of significant features after offspring mutation n_c and presumed target dimensionality d (see Fig. 2b). After local improvement terminates, all offspring genes whose corresponding attributes belong to the updated subset $\boldsymbol{\Xi}$ are set to one, the other are assigned zeros.

3.2 Constrained-Dimensionality HGA

The most intense argument against F-HGA is the requirement to determine dimensionality of target feature subspace prior to algorithm execution. However, feature subsets of some particular cardinality may not contain optimal or even suboptimal solution. Therefore, we propose another concept of the local search operator — apart from improving clustering quality, it allows finding optimal number of relevant features. In order to achieve that, it incorporates two extra procedures: conditional removal and addition (*rem_incr* and *add_incr* respectively — cf. Fig. 3a). These operations increase or decrease number of features in the current solution if only it improves objective function.

Each loop of *rem_incr* and *rem_incr* alters the size of $\boldsymbol{\Xi}$ (denoted by n_c) by one. Although the proposed local improvement mechanism does not demand fixing the value for target dimensionality d, lower and upper bounds have to be imposed ($d_{\min}$ and $d_{\max}$ respectively). Figure 3b depicts local improvement method in the C-HGA strategy.

In the above, we have made a tacit assumption that number of clusters K is known a priori. In order to determine it automatically, we propose to perform

```
Conditional addition (add_incr):
do
    $\hat{\delta} = \arg\max_{\delta \in \boldsymbol{\Delta}} J(\boldsymbol{\Xi} \cup \{\delta\})$;
    if $J(\boldsymbol{\Xi} \cup \{\hat{\delta}\}) > J(\boldsymbol{\Xi})$ then
        $\boldsymbol{\Xi} = \boldsymbol{\Xi} \cup \{\hat{\delta}\}; \boldsymbol{\Delta} = \boldsymbol{\Delta} \setminus \{\hat{\delta}\}$;
while$J(\boldsymbol{\Xi} \cup \{\hat{\delta}\}) > J(\boldsymbol{\Xi})$

Conditional removal (rem_incr):
do
    $\hat{\xi} = \arg\max_{\xi \in \boldsymbol{\Xi}} J(\boldsymbol{\Xi} \setminus \{\xi\})$;
    if $J(\boldsymbol{\Xi} \setminus \{\hat{\xi}\}) > J(\boldsymbol{\Xi})$ then
        $\boldsymbol{\Xi} = \boldsymbol{\Xi} \setminus \{\hat{\xi}\}; \boldsymbol{\Delta} = \boldsymbol{\Delta} \cup \{\hat{\xi}\}$;
while$J(\boldsymbol{\Xi} \setminus \{\hat{\xi}\}) > J(\boldsymbol{\Xi})$
```

(*a*)

```
$n_c = \|\boldsymbol{\Xi}\|$;
switch
    case $n_c < d_{\min}$
        for $i = 1$ to $(d_{\min} - n_c)$
            ripple_add(r);
        while $n_c < d_{\max}$
            add_incr;
    case $n_c = d_{\min}$
        while $n_c < d_{\max}$
            add_incr;
    case $n_c > d_{\min}$ and $n_c \leq d_{\max}$
        while $n_c > d_{\min}$
            rem_incr;
        while $n_c < d_{\max}$
            add_incr;
    case $n_c > d_{\max}$
        for $i = 1$ to $(n_c - d_{\max})$
            ripple_rem(r);
        while $n_c > d_{\min}$
            rem_incr;
```

(*b*)

Fig. 3. Pseudo-code of conditional addition and removal operations (a) and the local improvement operator in C-HGA (b)

several runs of HGA, each time with different presumption concerning K. Solution which gives the highest clustering quality over all runs indicates the most probable value of K.

4 Experiments

4.1 Experimental Framework

Evaluating effectiveness of feature space exploration methods involves considering different aspects of the unsupervised feature selection task. The comparison presented below examines three search strategies — both versions of HGA and NSGA — in 4 test areas 1) localizing feature subspaces ensuring high quality partitioning, 2) minimizing number of significant features, 3) finding optimal solution with high convergence rate, and 4) identifying correct number of clusters. In the case of NSGA, similarly to experiments reported by [7], there are two optimized objectives: clustering quality and feature space dimension. F-HGA is excluded from test area 2 (as it requires fixed target dimensionality) and also from test area 4 (its behaviour shall not remarkably differ from C-HGA in this respect). Independently from search strategy, clustering quality is evaluated using global silhouette value [3].

The data sets used in the experiments were constructed using a series of eight Brodatz album natural texture images (*tree bark, brick wall, beach sand, calf leather, herringbone weave, plastic bubbles, rafia and woolen clothe*) [10]. Every texture image was divided into 56 non-overlapping square-shaped regions of interest (ROI), each of the size 64 × 64 pixels. Next, all ROIs were submitted to texture analysis. This stage was accomplished by the help of MaZda software [11]. Each ROI was described by 270 texture parameters derived from co-occurrence and run-length matrices, autoregressive model, wavelet transform and absolute gradient. Feature vectors corresponding to different textures were then combined into various sets of 2, 3, 4 and 5 classes (10 data sets were formed in each case).

4.2 Results

Table 1 presents clustering quality of the best solutions (test area 1) obtained by the analyzed search strategies, averaged within each group of data sets. Independently from a data set, it was assumed that target dimension d (or its maximal value $d_{\max}$ in the case of C-HGA) equals to 10. It can be seen that C-HGA performs better than other techniques. Its supremacy reveals mainly for data sets containing more than two classes. The proposed strategy allows obtaining highest quality clusters within each data set group. NSGA in this aspect appears to be the worst choice.

In the test area 2, two NSGA solutions are taken into account. The first chromosome maximizes quality of clustering, while the second one minimizes number of relevant features. The averaged results obtained for each class-group and depicted in Table 2 show that NSGA is capable of finding lowest-dimensional feature subspaces. However, if the clustering quality is concerned, dimensions of the best NSGA solutions surpass the size of C-HGA-selected subsets.

In order to estimate computational effort (test area 3) associated with each of analyzed evolutionary search strategies we measure number of generations necessary to achieve final solution. The averaged results, gathered in Table 3 show that both C-HGA and F-HGA are superior to NSGA.

In the test area 4 we analyze two approaches which will be referred to as mode I and mode II. Mode I refers to the idea of identifying number of clusters simultaneously with features relevance by encoding K on dedicated chromosome genes, as described by [5]. On the other hand, mode II realizes the concept which

Table 1. Global silhouette values averaged over data sets within class-groups

K^a	NSGA	F-HGA	C-HGA
2	0,93	0,94	0,95
3	0,89	0,92	0,93
4	0,86	0,88	0,89
5	0,83	0,87	0,89

[a] K — number of classes in a data set group.

Table 2. Final feature space dimension and global silhouette values averaged over all data sets within each class-group

K[a]	NSGA-N[b]		NSGA-SV[c]		C-HGA	
	N	Υ	N	Υ	N	Υ
2	2,5	0,89	6,6	0,93	1,9	0,95
3	3,0	0,85	5,9	0,89	2,8	0,93
4	4,8	0,83	9,0	0,86	5,4	0,89
5	4,7	0,78	8,6	0,83	6,1	0,89

[a] K — number of classes in a data set group.
[b] NSGA-N — NSGA solution best at feature space dimension.
[c] NSGA-SV — NSGA solution best at clustering quality.

Table 3. Average number of generations necessary to achieve optimal solution

K[a]	NSGA	F-HGA	C-HGA
2	31	15	11
3	54	15	11
4	86	28	15
5	130	35	16

[a] K — number of classes in a data set group.

involves running feature selection with K set a priori separately for every K from the feasible range of values. We apply mode II to C-HGA and NSGA strategies, while mode I is used only with NSGA. Table 4 shows percentage of accurate estimates obtained by the analyzed methods. It can be observed that C-HGA gains the best results for each data set class-group, whereas NSGA in mode I achieves the poorest performance. NSGA fails since optimization procedure can be biased towards irrelevant feature subsets if for a given set of attributes wrong number of clusters ensures better estimates of clustering quality than some true partitioning.

Table 4. Percentage of accurately determined number of clusters [%]

K[a]	Mode I	Mode II	
	NSGA	NSGA	C-HGA
2	100	100	100
3	70	80	100
4	50	80	100
5	30	70	90

[a] K — number of classes in a data set group.

5 Conclusions

In this study we introduced a novel hybrid genetic algorithm (HGA) for unsupervised feature selection. We call our approach constrained-dimensionality HGA (C-HGA), as it searches for feature subsets whose size falls into the predefined range of feasible target dimensions. In contradiction to the strategy previously described in the literature [9], to which we refer as fixed-dimensionality HGA (F-HGA), local sequential search operator is effective only when it improves evaluation function. Otherwise, candidate feature subsets remain unchanged. Experimental results prove robustness of the proposed approach in various aspects of unsupervised knowledge discovery.

Acknowledgements

This work was supported by the Polish Ministry of Science and Higher Education grant no. 1205/DFG/ 2007/02.

References

1. Kohavi, R., John, G.: Wrappers for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
2. Davies, D., Bouldin, W.: A cluster separation measure. IEEE Trans. Pattern Analysis and Machine Intelligence 1(4), 224–227 (1979)
3. Struyf, A., Hubert, M., Rousseeuw, P.: Integrating robust clustering techniques in s-plus. Computational Statistics & Data Analysis 26, 17–37 (1997)
4. Siedlecki, W., Sklansky, J.: On automatic feature selection. International Journal of Pattern Recognition and Artificial Intelligence 2(2), 197–220 (1988)
5. Kim, Y., Street, W., Menczer, F.: Evolutionary model selection in unsupervised learning. Intelligent Data Analysis 6, 531–556 (2002)
6. Goldberg, D.: Genetic Algorithms in Search Optimization and Machine Learning. Addison-Wesley, Reading (1989)
7. Morita, M., Sabourin, R., Bortolozzi, F., Suen, C.: Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition. In: Seventh IEEE Conf. Document Analysis and Recognition, pp. 666–670 (2003)
8. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation 3(2), 221–248 (1994)
9. Oh, I., Lee, J., Moon, B.: Hybrid genetic algorithms for feature selection. IEEE Trans. Pattern Analysis and Machine Intelligence 26(11), 1424–1437 (2004)
10. Brodatz, P.: Textures. A Photographic Album for Artists and Designers. Dover, New York (1966)
11. Szczypinski, P., Strzelecki, M., Materka, A., Klepaczko, A.: Mazda - a software package for image texture analysis. Comput. Methods Prog. Biomed. 94, 66–76 (2008)

An Evolutionary Algorithm for Global Induction of Regression Trees

Marek Krętowski and Marcin Czajkowski

Faculty of Computer Science
Bialystok University of Technology
Wiejska 45a, 15-351 Białystok, Poland
{m.kretowski,m.czajkowski}@pb.edu.pl

Abstract. In the paper a new evolutionary algorithm for induction of univariate regression trees is proposed. In contrast to typical top-down approaches it globally searches for the best tree structure and tests in internal nodes. The population of initial trees is created with diverse top-down methods on randomly chosen sub-samples of the training data. Specialized genetic operators allow the algorithm to efficiently evolve regression trees. The complexity term introduced in the fitness function helps to mitigate the over-fitting problem. The preliminary experimental validation is promising as the resulting trees can be significantly less complex with at least comparable performance to the classical top-down counterpart.

1 Introduction

Regression is beside classification the most common predictive task in data mining applications [4]. It relies on associating numeric outcomes to objects based on their feature values. Regression trees are now popular alternatives to classical statistical techniques like standard regression or logistic regression [6]. The popularity of the tree-based approaches can be explained by their ease of application, fast operation and what may be the most important, their effectiveness. Furthermore, the hierarchical tree structure, where appropriate tests from consecutive nodes are sequentially applied, closely resembles a human way of decision making which makes regression trees natural and easy to understand even for not experienced analyst.

There exists a lot of regression tree systems. One of the first solutions was presented in the seminal book describing the *CART* system [2]. *CART* uses the sum of squared residuals as an impurity function and builds a piecewise-constant model with each terminal node fitted by the training sample mean. A lot of effort was then placed in developing model trees, which extend standard regression trees by replacing in leaves single predicted values by more advanced models (e.g. linear). *M5* [13] proposed by Quinlan, *RT* [14] developed by Torgo or *SECRET* [3] are good examples of such model trees. Recently, another model tree system *SMOTI* [11] was introduced and it enables associating multiple linear

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 157–164, 2010.
© Springer-Verlag Berlin Heidelberg 2010

model with a leaf by combining straight-line regressions reported along the path from the root.

It should be noticed that all aforementioned systems induce regression trees in a top-down manner. Starting from the root node they search for the locally optimal split (test) according to the given optimality measure and then the training data is redirected to newly created nodes. This procedure is recursively repeated until the stopping criteria is not met. Finally, the post-pruning is applied to improve the generalization power of the predictive model. Such a greedy technique is fast and generally efficient in many practical problem, but obviously does not guarantee the globally optimal solution. It can be expected that more global induction could be more adequate in certain situations.

In this paper we want to investigate a global approach to regression tree induction based on a specialized evolutionary algorithm. In our previous works we showed that evolutionary inducers are capable to efficiently induce various types of classification trees: univariate [8], oblique [9] and mixed [10]. In this paper we want to show that similar approach can be applied to obtain accurate and compact regression trees.

The rest of the paper is organized as follows. In the next section a new evolutionary algorithm for global induction of univariate regression trees is described. Experimental validation of the proposed approach on artificial and real-life data is presented in section 3. In the last section, the paper is concluded and possible future works are sketched.

2 Evolutionary Induction of Regression Trees

The general structure of the system follows a typical framework of evolutionary algorithms [12] with an unstructured population and a generational selection.

2.1 Representation, Initialization and Termination Condition

Representation. In the presented system, regression trees are represented in their actual form as classical univariate trees[1]. Each test in a non-terminal node concerns only one attribute (nominal or continuous valued). Additionally, in every node information about learning vectors associated with the node is stored. This enables the algorithm to perform more efficiently local structure and tests modifications during applications of genetic operators. Every leaf is associated with the predicted value estimated as a mean of dependent variable values from training objects in this leaf.

In case of a nominal attribute at least one value is associated with each branch. It means that an inner disjunction is built-in into the induction algorithm. For a continuous-valued feature typical inequality tests are applied. As potential splits only precalculated candidate thresholds are considered. A candidate threshold for the given attribute is defined as a midpoint between such a successive pair

[1] Tree-based representation is typical for genetic programming and for the first idea of evolving decision trees was proposed by Koza [7].

of examples in the sequence sorted by the increasing value of the attribute, in which the examples are characterized by different predicted values. Such a solution significantly limits the number of possible splits and focuses the search process.

Initialization. In the proposed approach, initial individuals are created by applying the classical top-down algorithm to randomly chosen sub-samples of the original training data (arbitrary default: 10% of data, but not more than 500 examples). Additionally, for every initial tree one of three test search strategies in non-terminal nodes is applied. Two strategies come from the very well-known regression tree systems i.e. *CART* [2] and *M5* [13] and they are based on *Least Squares* or *Least Absolute Deviation.* The last strategy is called *dipolar*, where a pair of feature vectors (dipole) is selected and then a test is constructed which splits this dipole. Selection of the dipole is randomized but longer (with bigger difference between dependent variable values) dipoles are preferred and mechanism similar to the ranking linear selection [12] is applied. The recursive partitioning is finished when all training objects in a node are characterized by the same predicted value, the number of objects in a node is lower than the predefined value (default value: 5) or the maximum tree depth is reached (default value: 10).

Termination condition. The evolution terminates classically when the fitness of the best individual in the population does not improve during the fixed number of generations [12] (default value is equal 1000). Additionally maximum number of generations is specified, which allows limiting the computation time in case of a slow convergence (default value: 5000).

2.2 Genetic Operators

There are two specialized genetic operators corresponding to the classical mutation and cross-over. Application of both operators can result in changes of the tree structure and tests in non-terminal nodes. After applying any operator it is usually necessary to relocate learning vectors between parts of the tree rooted in the altered node. This can cause that certain parts of the tree does not contain any learning vectors and has to be pruned.

Mutation operator. A mutation-like operator is applied with a given probability to a tree (default value is 0.8) and it guarantees that at least one node of the selected individual is mutated. Firstly, the type of the node (leaf or internal node) is randomly chosen with equal probability and if a mutation of a node of this type is not possible, the other node type is chosen. A ranked list of nodes of the selected type is created and a mechanism analogous to ranking linear selection is applied to decide which node will be affected. While concerning internal nodes, the location (the level) of the node in the tree and the quality of the subtree starting in the considered node are taken into account. It is evident that modification of the test in the root node affects whole tree and has a great impact, whereas mutation of an internal node in lower parts of the tree has only a local impact. In the proposed method, nodes on higher levels of the tree are mutated with lower probability and among nodes on the same level the absolute error calculated on the learning vectors located in the subtree is used to sort them. As for leaves, only absolute error is used to put them in order, but homogenous leaves are not included. As a result, leaves which are worse in terms of accuracy are mutated with higher probability.

Modifications performed by mutation operator depend on the node type (i.e. if the considered node is a leaf node or an internal node). For a non-terminal node a few possibilities exist:

- A completely new test can be found by means of the dipolar method used for the initialization;
- The existing test can be altered by shifting the splitting threshold (continuous-valued feature) or re-grouping feature values (nominal features);
- A test can be replaced by another test or tests can be interchanged;
- One sub-tree can be replaced by another sub-tree from the same node;
- A node can be transformed (pruned) into a leaf.

Modifying a leaf makes sense only if it contains objects with different dependent variable values. The leaf is transformed into an internal node and a new test is chosen in the aforementioned way.

Cross-over operator. There are also three variants of recombination. All of them start with selecting of cross-over positions in two affected individuals. One node is randomly chosen in each of two trees. In the most straightforward variant, the subtrees starting in the selected nodes are exchanged. This corresponds to the classical cross-over from genetic programming. In the second variant, which can be applied only when non-internal nodes are randomly chosen and the numbers of outcomes are equal, only tests associated with the nodes are exchanged. The third variant is also applicable only when non-internal nodes are drawn and the numbers of descendants are equal. Branches which start from the selected nodes are exchanged in random order.

2.3 Selection

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

2.4 Fitness Function

A fitness function drives the evolutionary search process and is very important and sensitive component of the algorithm. When concerning any prediction task it is well-known that the direct minimization of the prediction error measured on the learning set leads to an over-fitting problem. In a typical top-down induction of decision trees, the over-specialization problem is partially mitigated by defining a stopping condition and by applying a post-pruning. In our approach, the search for an optimal structure is embedded into the evolutionary algorithm by incorporating a complexity term into the fitness. This term works as a penalty for increasing the tree size.

The fitness function is minimized and has the following form:

$$Fitness(T) = \frac{MAE(T)}{MAE_{Max}} + \alpha \cdot (S(T) - 1.0), \tag{1}$$

where $MAE(T)$ - *Mean Absolute Error* of the tree T measured on the learning set and $S(T)$ - tree size. Subtracting 1.0 eliminates the penalty when the tree is composed of only one leaf. MAE_{Max} is the maximal $MAE(T)$ for the learning set. It can be easily calculated, as it is observed when the tree is composed of only one leaf. For any (more complex) tree T $MAE(T) \leq MAE_{Max}$. α is the relative importance of the complexity term (default value is 0.001) and a user supplied parameter.

It should be noticed that there is no optimal value of α for all possible datasets and tuning it may lead to the improvement of results for the specific problem.

3 Experimental Validation

Two groups of experiments are performed to validate the global approach to induction of regression trees (denoted in tables as *GRT*). For the purpose of comparison, results obtained by the classical top-down inducer *REPTree*, which is publicly available in the *Weka* system [5], are also presented. *REPTree* builds a regression tree using variance and prunes it using reduced-error pruning (with backfitting). Both systems were run with default values of parameters. All results presented in the table correspond to averages of 10 runs and were obtained by using test sets (when available) or by 10-fold cross-validation. The average number of nodes is given as a complexity measure of regression trees.

Synthetical datasets. In the first group, two simple artificially generated datasets with analytically defined decision borders are analyzed. Both datasets contain two independent features and one dependent feature with only a few distinct values. Number of feature vectors in the learning sets is 1000.

In figure 1 three-dimensional visualization of the datasets with the corresponding regression trees generated by the global method are presented. It should be noticed that in both cases trees obtained by *GRT* have optimal structure (only 9 and 4 leaves correspondingly) and gain very small error (RMSE equal to 0.139 for *chess* and 0.102 for *armchair*) on the testing set.

For the top-down inducer both problem are too difficult. For the first dataset *REPTree* generates an overgrown tree with 21 leaves and as a result the testing error is significantly higher (0.288). For the *armchair* problem the solution found by the *REPTree* is also not completely optimal (6 leaves and error=0.149), as the first split ($x < 2.02$) is not the best.

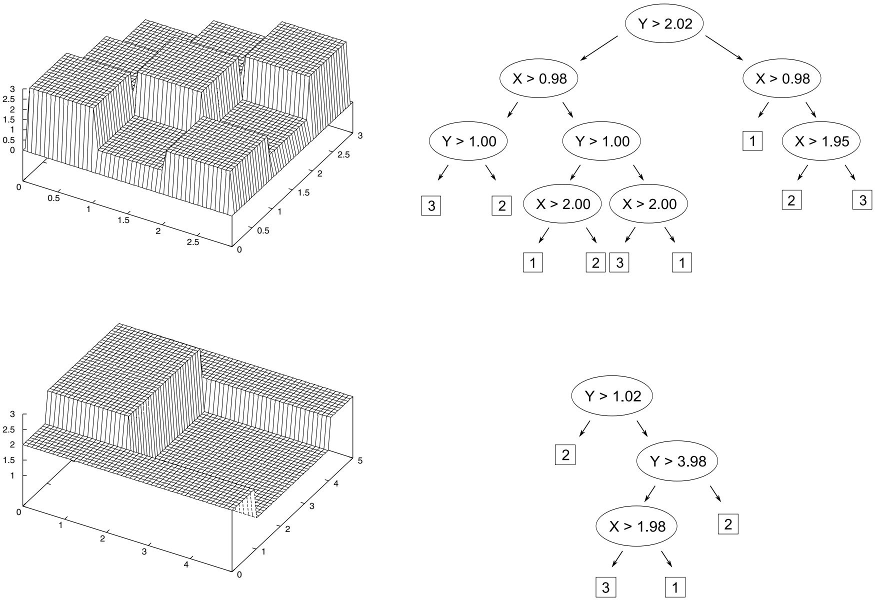


Fig. 1. Examples of artificial datasets (*chess* - top, *armchair* - bottom) and the corresponding regression trees generated by *GRT*

Real-life datasets. In the second series of experiments, several datasets taken from UCI Machine Learning Repository [1] or provided by L. Torgo on his website are analyzed to assess the performance of the proposed system in solving real-life problems. Table 1 presents characteristics of investigated datasets and obtained results.

It can be observed that the prediction accuracy of both analyzed system is comparable. It should be however noticed that globally induced trees can be significantly less complex. It is especially visible for the biggest trees generated by *REPTree*, where *GRT* is able to find smaller solutions (eg. 67.2 nodes as opposed to 819 nodes or 53 to 553).

It should be underline that the results collected in Table 1 were obtained with the default value of α parameter. In order to verify the impact of this parameter on the results, a series of experiments with varying α was prepared (see Fig. 2) on two exemplar datasets. As it could be expected, along with a decrease of α an increase of trees complexity can be observed. As for RMSE after initial quick decrease the error rate begins to rise slightly. It can be also observed that for both datasets the default setting of this parameter is not really optimal and smaller errors can easily obtained.

Table 1. Characteristics of the real-life datasets (number of objects/number of numeric features/number of nominal features) and obtained results. Root mean squared error (RMSE) is given as the error measure and number of nodes as the tree size.

		GRT		*REPTree*	
Dataset	Properties	RMSE	Tree size	RMSE	Tree size
Abalone	4177/7/1	2.30	49.3	2.36	201
Ailerons	13750/40/0	0.00022	53.3	0.00020	553
Auto-Mpg	392/4/3	3.59	145.6	3.65	94
Auto-Price	159/17/10	2505	91.2	2760	32
Delta Ailerons	7129/6/0	0.000182	29.8	0.000175	291
Delta Elevators	9517/6/0	0.00156	25.0	0.0015	319
Elevators	16559/40/0	0.0044	65.9	0.0040	503
Housing	506/14/0	4.29	88.6	4.84	41
Kinematics	8192/8/0	0.193	67.2	0.191	819
Machine CPU	209/7/0	60.4	75.8	92.3	15
Pole	15000/48/0	10.23	47.8	8.26	223
Pyrimidines	74/28/0	0.101	81.5	0.136	1
Stock	950/10/0	1.317	59.2	1.186	137
Triazines	186/61/0	0.149	159.0	0.152	7

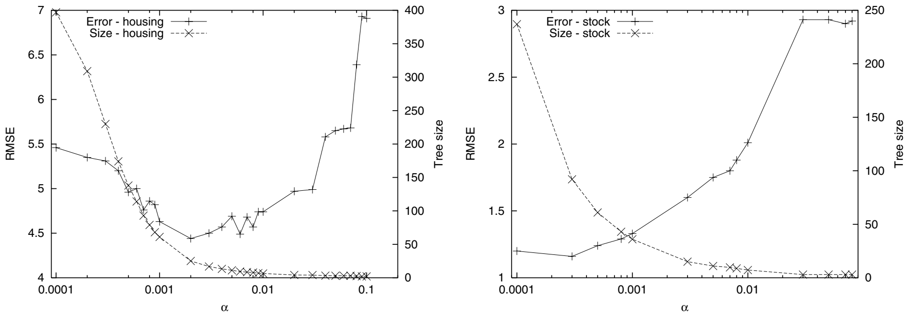


Fig. 2. Influence of α parameter on the performance of the *GRT* algorithm on two datasets

4 Conclusion

In the paper a new global approach to regression tree learning is presented. In contrast to classical top-down inducers, where locally optimal tests are sequentially chosen, both the tree structure and tests in internal nodes are searched in the same time by specialized evolutionary algorithm. This way the inducer is able to avoid local optima and to generate better predictive model. Even preliminary experimental results show that the globally evolved regression trees could be competitive compared to the top-down based counterparts, especially in term of tree size.

The presented approach is constantly improved and currently we are working on introducing oblique tests in the non-terminal nodes. On the other hand, we plan to extend the knowledge representation by evolving model trees.

Acknowledgments. This work was supported by the grant W/WI/5/10 from Bialystok University of Technology.

References

1. Blake, C., Keogh, E., Merz, C.: UCI repository of machine learning databases (1998), http://www.ics.uci.edu/~mlearn/MLRepository.html
2. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth Int. Group (1984)
3. Dobra, A., Gehrke, J.: SECRET: A scalable linear regression tree algorithm. In: Proc. KDD 2002 (2002)
4. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.): Advances in Knowledge Discovery and Data Mining. AAAI Press, Menlo Park (1996)
5. Frank, E., et al.: Weka 3 - Data Mining with Open Source Machine Learning Software in Java. University of Waikato (2000), http://www.cs.waikato.ac.nz/~ml/weka
6. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of statistical Learning. Data Mining, Inference, and Prediction, 2nd edn. Springer, Heidelberg (2009)
7. Koza, J.: Concept formation and decision tree induction using genetic programming paradigm. In: Schwefel, H.-P., Männer, R. (eds.) PPSN 1990. LNCS, vol. 496, pp. 124–128. Springer, Heidelberg (1991)
8. Krętowski, M., Grześ, M.: Global learning of decision trees by an evolutionary algorithm. In: Information Processing and Security Systems, pp. 401–410. Springer, Heidelberg (2005)
9. Krętowski, M., Grześ, M.: Evolutionary learning of linear trees with embedded feature selection. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 400–409. Springer, Heidelberg (2006)
10. Krętowski, M., Grześ, M.: Evolutionary induction of mixed decision trees. International Journal of Data Warehousing and Mining 3(4), 68–82 (2007)
11. Malerba, D., Esposito, F., Ceci, M., Appice, A.: Top-down induction of model trees with regression and splitting nodes. IEEE Trans. on PAMI 26(5), 612–625 (2004)
12. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs, 3rd edn. Springer, Heidelberg (1996)
13. Quinlan, J.: Learning with continuous classes. In: Proc. AI 1992, pp. 343–348. World Scientific, Singapore (1992)
14. Torgo, L.: Inductive learning of tree-based regression models. Ph.D. Thesis, University of Porto (1999)

Using Genetic Algorithm for Selection of Initial Cluster Centers for the *K*-Means Method

Wojciech Kwedlo and Piotr Iwanowicz

Faculty of Computer Science
Białystok University of Technology
Wiejska 45a, 15-351 Białystok, Poland
w.kwedlo@pb.edu.pl

Abstract. The K-means algorithm is one of the most widely used clustering methods. However, solutions obtained by it are strongly dependent on initialization of cluster centers. In the paper a novel genetic algorithm, called GAKMI (Genetic Algorithm for the K-Means Initialization), for the selection of initial cluster centers is proposed. Contrary to most of the approaches described in the literature, which encode coordinates of cluster centers directly in a chromosome, our method uses binary encoding. In this encoding bits set to one select elements of the learning set as initial cluster centers. Since in our approach not every binary chromosome encodes a feasible solution, we propose two repair algorithms to convert infeasible chromosomes into feasible ones. GAKMI was tested on three datasets, using varying number of clusters. The experimental results are encouraging.

1 Introduction

Clustering [12] is an unsupervised classification technique, which has applications in many areas, such as social sciences, biology, medicine and signal processing. The aim of clustering can be described as dividing a set of objects into K disjoint groups, called *clusters*, in such a way, that objects within one cluster are very similar and objects in different clusters are very distinct. Given the learning set of *feature vectors* $X = \{x_1, \dots, x_i \dots, x_N\}$, where $x_i \in \Re^d$ its partition G can be defined as $G = \{C_1, C_2, \dots, C_K\}$, where $\forall_{i \neq j} C_i \cap C_j = \emptyset$, $\bigcup_{i=1}^{K} C_i = X$, $\forall_i C_i \neq \emptyset$. The clustering problem can be then formulated as the problem of searching for a partition, which minimizes a certain criterion function. One of the most popular criterion functions is *sum of squared error* (SSE), which can be defined as:

$$\mathrm{SSE}(X, G) = \sum_{i=1}^{K} \sum_{x_j \in C_i} ||x_j - m_i||^2, \tag{1}$$

where m_i is the *center* of the cluster C_i, which can be computed as the sample mean.

The most popular clustering algorithm minimizing (1) is the K-means algorithm [7]. It is an iterative algorithm, which can be described by the following steps.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 165–172, 2010.
© Springer-Verlag Berlin Heidelberg 2010

1. Choose initial centers $\{m_1, \ldots, m_K\}$ of the clusters $\{C_1, \ldots, C_K\}$.
2. Calculate new cluster membership. A feature vector x_j is assigned to the cluster C_i if and only if

$$i = \operatorname*{argmin}_{k=1,\ldots,K} ||x_j - m_k||^2.$$

3. Recalculate the centers of the clusters.
4. If all the cluster centers have not changed in this iteration finish the algorithm. Otherwise goto Step 2.

The K-means algorithm is easy to implement and computationally efficient. However it has has an important deficiency. Although it converges in finite number of iterations, it can be easily trapped in a local minimum of the SSE function (1). Consequently, the quality of the solutions obtained by the algorithm is strongly dependent on the starting conditions (the initial cluster centers).

There are several methods for initializing the K-means algorithm. In the most popular approach referred to as the Forgy method [1], K randomly selected feature vectors serve as initial cluster centers. To alleviate somehow the problem of local minima the algorithm can be run many times, each time starting with different random set of feature vectors as the initial conditions. The result of the best (with the lowest SSE) run is reported as the result of the whole method. We call this approach Multiple Random Restart (MRR). Obviously MRR does not guarantee the globally optimal solution.

Some researchers suggest, that the performance of the K-means algorithm can be improved, if the initial centers are provided by another clustering method. Milligan [9] suggests the use of the Ward's hierarchical clustering algorithm [11] for that purpose. We call this method Ward+K-Means (WKM).

In the paper a novel approach for the selection of initial cluster centers for the K-means procedure is proposed. The approach is based on the Genetic Algorithm (GA) [5]. GAs, which belong to the broader class of the Evolutionary Algorithms (EAs), are stochastic search techniques inspired by the process of biological evolution. Unlike local optimization methods they simultaneously process a population of problem solutions, which gives them the ability to escape from local optima.

The idea of using an EA for the initialization of the K-means is not new. The main difference between our approach and those described in the literature (e.g. [3]) lies in the representation of the problem solution i.e. the set of initial cluster centers. Most of the methods use so called centroid-based representation [6], in which the coordinates of the cluster centers are encoded in a chromosome, usually a real-valued string of length dK, where d is the dimension of the feature space. The first d real numbers in a chromosome represent coordinates of the first cluster center, the next d numbers represent coordinates of the second cluster center and so forth. This method suffers from the so called redundancy problem, since each partition can be encoded by $K!$ different chromosomes.

In our approach, which we call GAKMI (GAKMI, for Genetic Algorithm for the K-Means Initialization), the set of initial cluster centers is represented by a binary string of length N, where N is the number of feature vectors. If and only if the i-th bit is set, then the feature vector x_i is used as a cluster center in the initialization of the K-means. Contrary to the centroid-based representation this method is not redundant. However it requires the modification of the standard GA, since only a chromosome with exactly K bits set encodes a feasible solution. Our modification consists in introduction of a *chromosome repair algorithm* [8], which is run on each infeasible chromosome before fitness evaluation.

2 Description of GAKMI

GAKMI is based on the standard scheme of Genetic Algorithm [5,8], in which multiple iterations of three consecutive steps (computation of the fitness function, selection and reproduction) are conducted. As selection we use the proportional method with the sigma truncation scaling [5], where the value of the scaling parameter is set to 2. Reproduction employs standard bit-flip mutation and two-point crossover.

2.1 Encoding of Solutions

A chromosome S represents a set of feature vectors $U = \{u_1, u_2, \ldots, u_K\}$, where $u_i \in X$. The K elements of U are used as the initial cluster centers for the run of the K-means algorithm, which is performed during the fitness evaluation. The chromosome encoding U is a binary string of length N, where N is the size of the learning set X. The i-th bit of the chromosome corresponds to the i-th feature vector x_i. Denote by the $S(i), i = 1, 2, \ldots, N$ the value of the $i - th$ bit in the chromosome. If and only if $S(i) = 1$, then $x_i \in U$. For instance, if $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$, then the chromosome $S = 100010$ represents the set $U = \{x_1, x_5\}$.

It is important to notice, that not every binary string represents a valid solution to the K-means initialization problem. Since we need exactly K initial centers i.e $|U| = K$, only a string with exactly K bits set and $N - K$ bits not set represents a *feasible* solution. Formally a chromosome S is feasible if and only if $\sum_{i=1}^{N} S(i) = K$. If this condition is not met the chromosome is called *infeasible*. It is possible, that an infeasible chromosome will appear in the population as a result of crossover and mutation operators. In such case the infeasible chromosome is converted into feasible one by a repair algorithm.

2.2 Fitness Function

Because the aim of the GA is to find the best initial conditions for the K-means algorithm, the most natural way for evaluating the chromosome S consists in running the K-means algorithm initialized by the subset of feature vectors represented by S. The K-means algorithm is run until the convergence and the fitness of the chromosome is based on the value of SSE (1) of the final clustering solution. Since the proportional selection method is based on the assumption of a maximization problem with nonnegative values of the fitness function whereas clustering problem with SSE as the criterion function is a minimization problem, the SSE of a chromosome is converted to its fitness according to the equation

$$f(S) = -\mathrm{SSE}(S) + \mathrm{SSE}_{\max} + \mathrm{SSE}_{\min}.$$

In this equation $f(S)$ is the fitness of the chromosome S, $\mathrm{SSE}(S)$ is the SSE obtained by the run of K-means initialized by feature vectors selected by S, $\mathrm{SSE}_{\max}$ and $\mathrm{SSE}_{\min}$ denote the maximal and minimal values of SSE, respectively, in the population of chromosomes.

2.3 Repair Algorithms

A repair algorithm is run before fitness evaluation. It is applied to each infeasible chromosome created by crossover and mutation operators. Then, the repaired chromosome replaces original one in the population. After the replacement of infeasible chromosomes by feasible ones the fitness evaluation is performed. Denote by S^+ and S^- sets of bit positions in a chromosome S, which have corresponding bits set and not set, respectively. Formally $S^+ = \{i : S(i) = 1\}$ and $S^- = \{i : S(i) = 0\}$.

Random Repair. The random repair algorithm, denoted in the paper as RR, flips randomly chosen bits in an infeasible chromosome in order to convert it into feasible one. If too few bits are set i.e. $|S^+| < K$, then $K - |S^+|$ bits at the positions randomly chosen from S^- are set. Otherwise, if too many bits are set i.e. $|S^+| > K$, $|S^+| - K$ bits at the positions randomly chosen from S^+ are unset.

Distance-Based Repair. The distance-based repair algorithm is denoted in the paper as DBR. When $|S^+| < K$, i.e. chromosome S represents too few centers, DBR adds new feature vectors to the set of centers, paying more attention to vectors, which are most far apart from the existing centers encoded by the chromosome. The rationale behind this approach is the possibility of discovering new clusters, which are not covered by the existing centers. The algorithm repeats $K - |S^+|$ times the following steps:

1. Generate randomly a set $X_r = \{x_{r_1}, x_{r_2}, \ldots, x_{r_t}\}$, where $x_{r_i} \in X \wedge r_i \in S^-$, t is a parameter of the algorithm (in all the experiments we used $t = 50$). The elements of X_r are the candidates for inclusion into the set of chromosome centers encoded by S. For each $x_{r_i} \in X_r$ compute the minimal squared distance $d_1(x_{r_i})$ to the centers represented by the chromosome S:

$$d_1(x_{r_i}) = \min_{j \in S^+} ||x_{r_i} - x_j||^2$$

2. Select as the a new center the feature vector x_{r_*} with the maximal minimal squared distance, i.e.

$$r_* = \operatorname*{argmax}_{k=r_1,r_2,\ldots,r_t} d_1(x_k).$$

   Set $S(r_*) = 1$, $S^+ = S^+ \cup \{r^*\}$, $S^- = S^- \setminus \{r^*\}$.

When $|S^+| > K$, i.e. chromosome represents too many centers, DBR prefers removal of centers, which are situated close to other centers represented by the chromosome. This approach is based on the assumption, that if two centers are placed close from one another, they are likely to belong to the same cluster and one of them can be removed from the set of initial centers. The removal of centers is performed by repeating $|S^+| - K$ times the following procedure.

1. Generate randomly a set $X_s = \{x_{s_1}, x_{s_2}, \ldots, x_{s_t}\}$, where $x_{s_i} \in X \wedge s_i \in S^+$. The elements of X_s are the candidates for removal from the set of chromosome centers encoded by S.
2. For each $x_{s_i} \in X_s$ compute the minimal squared distance $d_2(x_{s_i})$ to other centers (excluding x_{s_i}) represented by chromosome.
$$d_2(x_{s_i}) = \min_{k \in S^+ \wedge k \neq s_i} ||x_{s_i} - x_k||^2$$
3. Select for removal a feature vector x_{s_*} which is situated the most close from other centers, i.e.
$$s_* = \operatorname*{argmin}_{k=s_1,s_2,\ldots,s_t} d_2(x_k).$$
Set $S(s_*) = 0$, $S^+ = S^+ \setminus \{s_*\}$, $S^- = S^- \cup \{s_*\}$.

3 Experimental Results

In this section the results of experiments, in which two variants of GAKMI were compared to MRR and WKM initialization methods, are provided. Two versions of GAKMI differed only by the repair algorithm. The version, which used random repair is denoted as GAKMI-RR, whereas the version, which used distance-based repair is denoted as GAKMI-DBR. In all the experiments GAKMI used the population size equal to 50 and crossover probability equal to 0.9. The mutation probability was always set to $1/N$, which ensured that on average one bit in a chromosome was changed by the mutation operator. The GAKMI results reported in this section were obtained by performing an average over 30 independent runs.

The algorithms GAKMI and MRR were compared on the basis of equal CPU time i.e the number of K-means restarts in MRR was chosen in a way, which ensured, that the runtime of MRR was equal to the runtime of GAKMI. For the WKM method, which is a deterministic algorithm, i.e. there is no gain from multiple runs, a comparison on the basis of equal runtime was not possible.

In all the tables in this section the first column shows the number of clusters K, the second column presents the SSE of the solution obtained using GAKMI-DBR method, whereas the following columns show % errors of the other three methods relative to the GAKMI-DBR. The % error of a method m is computed as $(\mathrm{SSE_m} - \mathrm{SSE_{GAKMI\text{-}DBR}})/\mathrm{SSE_{GAKMI\text{-}DBR}} * 100$. A negative % error corresponds to the situation, in which the method m outperforms GAKMI-DBR.

The first experiment was performed on a synthetic dataset S1 described in the paper by Fränti and Virmajoki [4]. The dataset, which was generated from mixture of 15 Gaussians, consists of 5000 two-dimensional feature vectors. The data form 15 well-separated clusters. Table 1 shows the results of this experiment.

Table 1. Clustering results for the S1 dataset

K	GAKMI-DBR (SSE)	GAKMI-RR (%)	WKM (%)	MRR (%)
10	3.43913E+13	0.00	0.00	0.00
20	7.71325E+12	0.00	0.23	0.01
30	5.91630E+12	0.00	1.20	0.64
40	4.63544E+12	0.00	1.79	1.73
50	3.74154E+12	0.00	0.83	1.95
60	3.05558E+12	0.00	1.56	2.62
70	2.55656E+12	0.00	1.79	3.33
80	2.23762E+12	0.00	0.87	2.05
90	1.99427E+12	0.04	1.50	3.31
100	1.79620E+12	0.05	2.36	3.70
110	1.63415E+12	0.06	2.03	4.52
120	1.49530E+12	0.09	1.79	4.43
130	1.37413E+12	0.13	1.98	5.67
140	1.27004E+12	0.20	2.45	5.86
150	1.18188E+12	0.27	2.34	6.52
	Average % error	0.06	1.51	3.09

The dataset used in the second experiment comes from the TSP-LIB library [10]. It consists of 1060 two-dimensional feature vectors representing position of the points in a plane. The clustering results for this dataset are shown in Table 2.

Table 2. Clustering results for the TSP-LIB dataset

K	GAKMI-DBR (SSE)	GAKMI-RR (%)	WKM (%)	MRR (%)
10	1.75484E+09	0.00	0.30	0.00
20	7.91795E+08	0.00	3.29	0.04
30	4.81252E+08	0.01	4.06	0.14
40	3.41343E+08	0.03	4.11	0.79
50	2.55528E+08	0.04	4.41	2.22
60	1.97273E+08	0.06	4.31	3.79
70	1.58451E+08	0.09	3.38	4.89
80	1.28893E+08	0.06	1.83	6.58
90	1.10428E+08	0.18	2.41	8.29
100	9.63662E+07	0.27	4.17	8.75
110	8.48667E+07	0.29	3.22	7.30
120	7.55625E+07	0.57	3.90	12.58
130	6.75738E+07	0.67	4.92	14.88
140	6.11824E+07	0.81	4.84	15.30
150	5.60013E+07	0.99	3.76	15.87
	Average % error	0.27	3.53	6.76

The third experiment was performed on the image segmentation dataset obtained from the UCI machine learning repository [2]. This dataset consists of 2310 19-dimensional feature vectors. Each of them represents a pixel of a image. The 19 features were extracted from 3x3 surrounding of the pixel. Table 3 shows the results of this experiment.

Table 3. Clustering results for the segmentation dataset

K	GAKMI-DBR (SSE)	GAKMI-RR (%)	WKM (%)	MRR (%)
10	9.79519E+06	0.00	1.76	0.00
20	5.12830E+06	0.00	2.18	0.07
30	3.50776E+06	-0.02	1.76	7.35
40	2.74198E+06	0.25	2.93	9.66
50	2.22782E+06	1.06	2.02	12.76
60	1.87506E+06	1.70	2.63	16.08
70	1.61864E+06	3.37	3.30	20.89
80	1.41782E+06	3.08	2.25	23.40
90	1.26129E+06	9.16	1.56	28.45
100	1.13676E+06	6.77	2.40	31.01
110	1.03436E+06	12.83	1.64	30.45
120	9.49088E+05	10.28	0.83	37.14
130	8.73945E+05	9.19	1.11	43.92
140	8.10890E+05	14.52	0.97	44.21
150	7.54956E+05	13.67	1.54	49.25
Average % error		5.72	1.92	23.64

Tables 1–3 suggest the following conclusions:

- Overall, GAKMI-DBR is the best K-means initialization method. In all cases, except one (the segmentation dataset, for $K = 30$), its performance was better or equal than the performance of other three methods.
- When the number of clusters is large enough, the choice of a repair algorithm has a considerable impact on the performance of our method. In such situations the distance-based repair should be preferred.
- For small number of clusters the MRR method achieves better results than the WKM method. For large number of clusters the opposite is true.

4 Conclusions and Future Work

In the paper we have described GAKMI, a new method based on a GA, for the initialization of the K-means algorithm. The novelty of our method lies in the representation of problem solutions by binary chromosomes, which select the elements of the learning set to serve as initial cluster centers. Two version of GAKMI, using two different repair algorithms were evaluated experimentally. The results, show that GAKMI-DBR, the version using the distance-based repair, is able to outperform two standard methods for initialization of the K-means: Multiple Random Restart using the Forgy approach and initialization by the Ward's hierarchical clustering method.

Several directions of the future research exist. One of them is development of a parallel version of GAKMI. Currently the most of the runtime of the algorithm is spent on the computation of the fitness function, which requires a run of the K-means algorithm. It would be straightforward to parallelize this step, by performing the fitness evaluation concurrently on processors of a parallel system.

Currently GAKMI uses the standard mutation and crossover, which may introduce infeasible chromosomes into the population. We are planning to developed specialized crossover and mutation operators, which would always produce feasible chromosomes. In this way the need for a repair algorithm in GAKMI would be eliminated.

Acknowledgment. The software for this work used the GAlib genetic algorithm package, written by Matthew Wall at the Massachusetts Institute of Technology. This work was supported by the grant S/WI/2/2008 from Białystok University of Technology.

References

1. Anderberg, M.R.: Cluster Analysis for Applications. Academic Press, New York (1973)
2. Asuncion, A., Newman, D.J.: UCI machine learning repository (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html
3. Babu, G.P., Murty, M.N.: A near-optimal initial seed value selection in k-means algorithm using a genetic algorithm. Pattern Recognition Letters 14(10), 763–769 (1993)
4. Fränti, P., Virmajoki, O.: Iterative shrinking method for clustering problems. Pattern Recognition 39(5), 761–775 (2006)
5. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading (1989)
6. Hruschka, E.R., Campello, R.J.G.B., Freitas, A.A., de Carvalho, A.C.P.L.F.: A survey of evolutionary algorithms for clustering. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 39(2) (2009)
7. McQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
8. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer, Heidelberg (1996)
9. Milligan, G.W.: An examination of the effect of six types of error perturbation on fifteen clustering algorithms. Psychometrika 45(3), 325–342 (1980)
10. Reinelt, G.: A traveling salesman problem library. ORSA Journal on Computing 3(4), 376–384 (1991)
11. Ward, J.H.: Hierarchical grouping to optimize an objective function. Journal of American Statistical Associacion 58(301), 236–244 (1963)
12. Xu, R., Wunsch, D.: Survey of clustering algorithms. IEEE Transactions on Neural Networks 16(3), 645–678 (2005)

Classified-Chime Sound Generation Support System Using an Interactive Genetic Algorithm

Noriko Okada[1], Mitsunori Miki[2], Tomoyuki Hiroyasu[3], and Masato Yoshimi[2]

[1] Graduate School of Engineering, Doshisha University
[2] Department of Science and Engineering, Doshisha University
[3] Department of Life and Medical Sciences, Doshisha University
1-3 Tatara Miyakodani Kyotanabe-shi, Kyoto, Japan, 610-0321
{nokada@mikilab,mmiki@mail,tomo@is,myoshimi@mail}.doshisha.ac.jp

Abstract. This research proposes a chime sound generation support system to readily generate intercom chime sounds that are agreeable to individual persons and to associate the chime sounds with visitors. In the proposed system, an interactive genetic algorithm (IGA) is used. Based on the melodies created by users, chime sounds are automatically generated in accordance with rules. The effectiveness of the proposed system is verified by an experiment using the system.

Keywords: Optimization, Interactive Genetic Algorithm, Chime sound.

1 Introduction

Most information is displayed by visual media in our daily lives. Recently, however, electronic devices are becoming smaller and more portable. The display of visual information is restricted, because the size of visual displays is limited [1]. At the same time, auditory information is not restricted by the display size. Accordingly, it is not necessary to pay attention visually to the object and it is possible to obtain auditory information in a paratactic manner, making information propagation highly effective [2].

In this background, alarms using sound are becoming quite common in home electric appliances. These auditory signals are often monotonous, however and not all users like them. Therefore, it may be useful for users to create their own melodies and use them as auditory signals. In addition, products that identify individuals and classify them into categories using sounds have also increased in recent years. For example, cell phone ringtones can be changed depending on the caller, allowing the user to know who a caller is before answering the phone and deal with the call in an appropriate manner. For intercoms, it is useful to be informed of visitors by sound, as well as be informed of who a visitor is before greeting them.

Visitors can be discriminated by using the latest intercom image processing technology. Therefore, it is useful to compose and set different melodies for individual visitors according to the user's taste, but this is difficult for people who

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 173–180, 2010.
© Springer-Verlag Berlin Heidelberg 2010

do not know how to compose melody. It may impose a tremendous burden on users to compose multiple chime sounds for visitors.

Therefore, this research proposes a chime sound generation support system that generates chime sounds suiting individual tastes, recalling individual visitors and bearing sounds with information that conveys a message using Interactive Genetic Algorithm (IGA) [3], one of the possible optimization methods. The authors of this paper have already developed a system to generate auditory signals using IGA [4]. This research applies this approach for the generation of auditory signals to the generation of chime sounds for intercoms and further proposes a method to change chimes depending on the visitor category.

2 Interactive Genetic Algorithm

An Interactive Genetic Algorithm (IGA) [3] is a Genetic Algorithm (GA) [5] which simulates evolution of organisms, where the evaluation part of the GA is handled subjectively by a human being. In problems which cannot be numerically quantified because they involve the impressions and tastes of human beings, optimization is done based on evaluation according to human sensibility. IGA is used for ”3-D CG Lighting” [6], ”Application of fashion design” [7] etc. In this research, the aim is using IGA which allows even ordinary users, who cannot create melodies using musical instruments, to simply create good chime sounds simply by evaluating several candidate solutions based on their own subjectivity. In this research, the aim is to develop a method based on IGA which allows even ordinary users, who cannot create melodies using musical instruments, to simply create good and purposeful sign sounds simply by evaluating several candidate melodies based on their own subjectivity.

3 Chime Sound Generation Support System

3.1 Overview of Chime Sound Generation Support System

This system generates chime sounds matching user tastes by having users evaluate proposed melodies. IGA is used for the method. Furthermore, based on the chime sounds prepared by a user, multiple chime sounds are automatically generated depending on the predetermined visitor category. Fig. 1 shows a conceptual diagram of the system.

3.2 Category Classification

In the preparatory experiment of this system, the appropriateness of the number of visitor categories necessary for the automated generation of melodies and the importance of the visitor discrimination criteria were dealt with. As a result, visitors were categorized into three categories, specifically family members, acquaintances and strangers. For category classification, visitors are classified by face recognition using the a camera in the intercom. Acquaintances are persons other than family members whose faces are registered in advance. Strangers are persons with no face data registered.

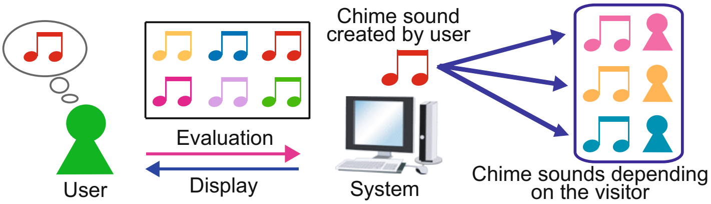


Fig. 1. Conceptual diagram of the chime sound generation support system

3.3 Method of Representing Sound

The method of representing sound in the proposed system was determined as follows by performing preliminary experiments from various standpoints, and taking those results into consideration. In this system, one melody is expressed as one individual. The notes which constitute a melody are quater note, quater rest, eighth note, and eighth rest. The note length was determined by taking one eighth note to be the basis for one note in this system. The length of the melody was made into 2 bars by 3/4 meter, tempo set to 125 Beats Per Minute, and the tone was taken as Vibraphone. The value expressing a quater note, and the value of note pitch are stored in each gene. Note numbers defined using a Standard MIDI File (SMF) were used to represent tone pitch. In the note numbers in the SMF, 60 is taken to be middle C(C4) on the piano, and the numbers change by 1 for each semi-tone. The lowest tone is defined to be 0, and the highest tone is defined to be 127. The note numbers used in the research were selected from the C major scale, from which it is easy to generate melodies with a bright sound, and the range of notes was set to note numbers 60 to 79 only. The rest was defined as 128 which is not used by SMF, and the value expressing a quater note was defined as 129 which is not used by SMF. Fig. 2 shows the relation among tone pitches, note numbers and pitch names, and Fig. 3 shows the chromosome structure.

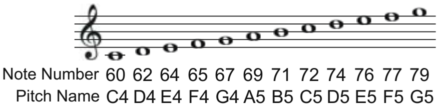

Note Number 60 62 64 65 67 69 71 72 74 76 77 79
Pitch Name C4 D4 E4 F4 G4 A5 B5 C5 D5 E5 F5 G5

Fig. 2. Relation among tone pitches, note numbers and pitch names

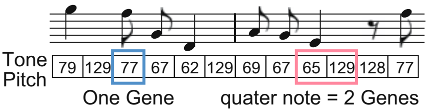


Fig. 3. Chromosome structure

3.4 Flow of Chime Sound Generation Support System

Fig. 4 shows the flow of melody generation in the system. The processes performed in each block in Fig. 4 are as follows:

1. Generation of initial individuals
   In IGA, optimized individuals are likely to depend on the initial individual.

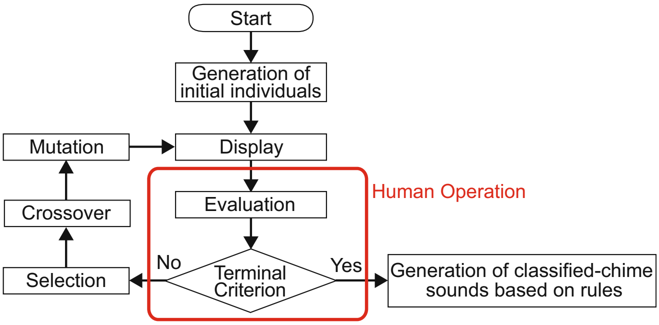


Fig. 4. Flow of chime sound generation support system

Therefore, it is important to propose as many initial individuals as possible. In this system, however, the number of individuals proposed at one time is limited. As a result, this system uses the following method to propose as many individuals as possible to users at an earlier stage.

A user evaluates 12 individuals randomly generated by the system using a five point scale. The system chooses four individuals with higher scores and adds two individuals newly generated in a random manner. The six individuals become the initial individual group. Individuals randomly generated have tone values that are randomly determined within the range of definition.

2. Display
   The system displays individuals as score corresponding to a melody like Fig. 5 to a user. User can listen to melody by pushing Play button.

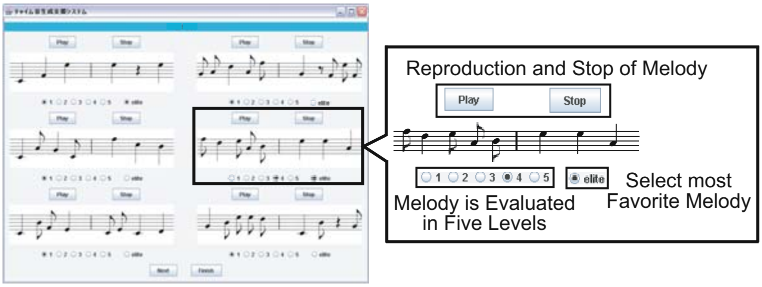

Reproduction and Stop of Melody
Play
Stop
○1 ○2 ○3 ●4 ○5
● elite
Melody is Evaluated in Five Levels
Select most Favorite Melody
Next
Finish

Fig. 5. Display

3. Evaluation
   The user listens to the melody, and evaluates it with a score of 1 to 5 points. Also, user chooses one individual as the ”elite individual”.
4. Selection
   The system is performes the designated selection (Roulette Selection and Elite Preservation) based on evaluation conducted by the user.

5. Crossover
   Crossover is performed so that the phrase in a melody may not be destroyed. Therefore, phrase size was set to one bar, and one-point crossover was performed in phrase units. Fig. 6 shows an example of crossover.

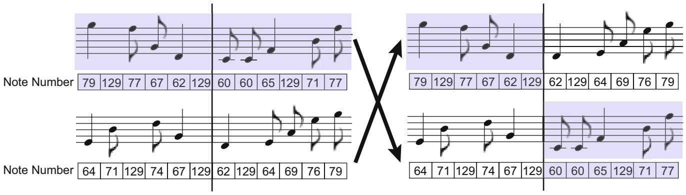


Fig. 6. Crossover

6. Mutation
   Mutation is performed to tone pitches. When the object of mutation is rest, tone pitch is varied randomly within the defined range. When it is not rest, if variation is done uniformly without designating a range for tone pitch, melody is uncomfortable, and thus variation is done randomly in a range of three steps above or below the height of the original tone.
7. Terminal criterion
   Search is ended when the chime sound which a user satisfies is able to be created. The minimum number of generations is 3 and a maximum is 10.
8. Generation of classified-chime sounds based on rules
   Based on the melodies created by users in operations 1 through 7, three types of melodies are generated based on the rules. A preliminary experiment was performed to determine the appropriateness of the rules for each category.
   - Family members
     The tone of the last three notes of the base melody is increased gradually using the sounds of the primary triads of C major.
   - Acquaintances
     The tones of all the notes in the base melody are unchanged except that they are changed into eighth notesD

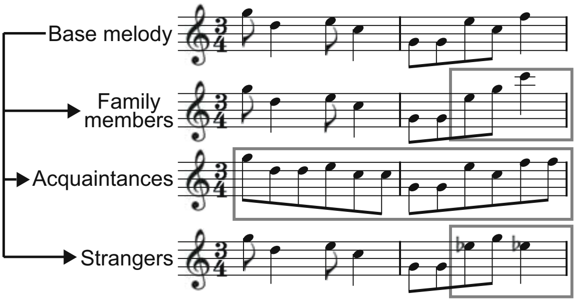


Fig. 7. Example of the rules

- Strangers
  The primary triads of C minor are used for the tones of the last three sounds of the base melody. The same tones are used for the last note and the third last note, while higher tones are used for the second last note.

Fig. 7 shows an example of the rules. The frames shown in Fig. 7 indicate the parts changed from the base melody.

4 Evaluation Experiment

4.1 Overview of the Experiment

An experiment was performed to verify the effectiveness of the proposed system. The subjects included 20 males and females in their 20s. In the experiment, the number of individuals in one generation was set to six, the crossover rate was set to 1.0 and the mutation rate was set to 0.16. In the experiment, subjects were ordered to make chime sounds using the proposed system. After that, a questionnaire was performed regarding items (1) through (3) shown below. Furthermore, subjects were asked to listen to chime sounds generated by the proposed system that varied by visitor and were asked to answer questionnaire items (4) and (5). The system generated melodies for the three categories of family members, acquaintances and strangers. Subjects were blinded to each melody category.

mQuestionnairen

(1) Do you have any knowledge regarding music composition, etc.?
(2) How satisfied are you with the chime sounds generated by the system?
(3) How easy do you think it is to make chime sounds using the system?
(4) Can you guess which melody is for family members, acquaintances and strangers, respectively?
(5) How much do you like each melody?

4.2 Experimental Results and Discussion

Fig. 8 through Fig. 12 show the results of questionnaire items (1) through (5), respectively.

Fig. 8 shows that most of the subjects had no knowledge of music such as composition. Fig. 9 shows that most of the subjects thought that they could satisfactorily compose melodies using the system. Furthermore, in Fig. 10, most of the subjects answered that it was easy to make melodies. As shown above, even persons without knowledge of music could easily make satisfactory melodies.

As shown in Fig. 11, when a melody targeted for ”family members” is changed, 55% of subjects determined that the changed melody was for ”family members,” while 45% of subjects determined that the changed melody was for ”acquaintances.” Discrimination between ”family members” and ”acquaintances” was not easy. It is believed that the familiarity of a melody is one element in determining the type of visitor that was different among subjects. Meanwhile, for

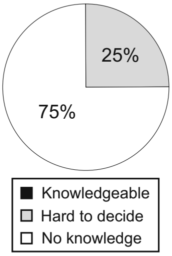


Fig. 8. Result of questionnaire item (1)

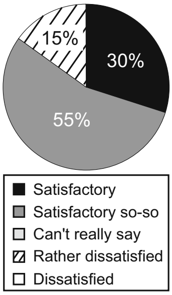


Fig. 9. Result of questionnaire item (2)

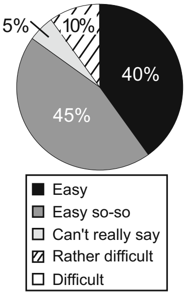


Fig. 10. Result of questionnaire item (3)

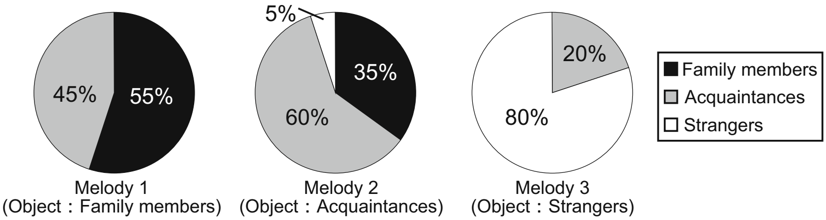


Fig. 11. Result of questionnaire item (4)

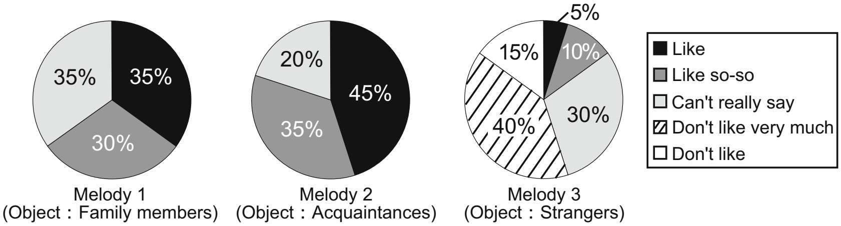


Fig. 12. Result of questionnaire item (5)

melodies targeted for "acquaintances" and "strangers" that were changed, more than 60% of subjects correctly determined the discrimination of the melody. It can be said that information that conveys a message was added to the melodies to discriminate visitors based on the rules.

Fig. 12 shows that the favorability rating of changed melodies for "family members" and "acquaintances" was very high. Meanwhile, subjects did not like changed melodies for "strangers." It is believed that the subjects did not like these melodies because the modified characteristics of the melody gave them an uncomfortable feeling.

As shown above, the basic melodies made by the system were satisfactory. Accordingly, rules for the melodies for each visitor category should be improved.

Specifically, it is necessary to improve familiarity for melodies for "family members" in order to discriminate between melodies for "family members" and "acquaintances." In respect to "strangers" on the other hand, information that conveys a message in the melody should remain and the rules for changing the melodies should be reviewed so that the melodies for "strangers" better suit user tastes.

5 Conclusion

In this research, a chime sound generation support system where favorite chime sounds can be created for an intercom and automatically generated depending on the category of visitor was constructed. The evaluation experiment showed that even users who had no knowledge about music could make satisfactory chime sounds easily by using the proposed system. In addition, it showed that melodies with information that conveys a message could be made for acquaintances and strangers and melodies for family members and acquaintances that highly suit the subjects' taste could be generated by the system based on the proposed rules.

References

1. Gaver, W.: The Sonic Finder: An Interface That Uses Auditory Icons. Human-Computer Interaction 4, 67–94 (1989)
2. Kramer, G.: An Introduction to Auditory Display. In: Kramer, G. (ed.) Auditory Display. Sonification, Audification and Auditory Interfaces, pp. 1–77. Addison-Wesley Pub. Co., Reading (1994)
3. Takagi, H.: Interactive evolutionary computation: fusion of the capabilities of ec optimization and human evaluation. Proceedings of the IEEE 89(9), 1275–1296 (2001)
4. Miki, M., Orita, H., Wake, S., Hiroyasu, T.: Design of Sign Sounds using an Interactive Genetic Algorithm. In: International Conference on Systems, Man and Cybernetics of the IEEE, pp. 3486–3490 (2006)
5. Goldberg, D.: Genetic Algorithms in Serch Optimization and Machine Learnig. Addision Wesley, Reading (1989)
6. Aoki, K., Takagi, H.: 3-D CG Lighting with an Interactive GA. In: 1st Int'l Conf. on Conventional and Knowledge-based Intelligent Electronic Systems, pp. 296–301 (1997)
7. Kim, H.-S., Cho, S.-B.: Application of interactive genetic algorithm to fashion design. Engineering Applications of Artificial Intelligence 13(6), 635–644 (2000)

Evolutionary Algorithms with Stable Mutations Based on a Discrete Spectral Measure

Andrzej Obuchowicz and Przemysław Prętki

Institute of Control and Computation Engineering
University of Zielona Góra,
ul. Podgórna 50, 65–246 Zielona Góra, Poland
A.Obuchowicz@issi.uz.zgora.pl,
P.Pretki@issi.uz.zgora.pl

Abstract. In this paper the concept of multidimensional discrete spectral measure is introduced in the context of its application to real-valued evolutionary algorithms. The notion of discrete spectral measure makes possible to uniquely define a class of multivariate heavy-tailed distributions, that have received more and more attention of evolutionary optimization commynity , recently. Simple sample illustrates advantages of such approach.

1 Introduction

Evolutionary algorithms (EAs) have been successfully applied to global optimization problems in many areas of engineering [11]. Their advantage over many other optimization techniques consists in the fact that EAs are based on only function evaluations and comparisons [6]. Thus, EAs are able to deal successfully with problems that cannot be easily solved by standard optimization procedures. Unfortunately, the EAs also suffer from many serious drawbacks. The most severe one is related with the appropriate choice of their control parameters, which to a large extent determine their performance. Usually, control parameters such as strength of mutation, population size, the selective pressure are chosen during trial-and-error process or on the base of expert knowledge, which, unfortunately, in usually inaccessible or the cost of its collection exceeds decidedly the computational cost of the optimization process itself. One way out of this difficulties is applying algorithms, which make use of some heuristics and dedicated techniques that aim at adjusting some of the control parameters automatically during optimization process.

In spite of the fact that the problem of parameters adaptation has been attacked from various angles by many authors and a number of relevant results have already been reported in the literature, there is still a lack of an unified theory that addresses the problem.

In the case of the EAs, most attention has been directed toward a normal distribution so far. Thus, several relevant approaches to adaptation of its parameters have already reported in the literature [1,5,8]. On the other hand, it

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 181–188, 2010.
© Springer-Verlag Berlin Heidelberg 2010

is noticeable that a normal distribution does not guarantee the highest performance of EAs, so that other distributions have aroused evolutionary algorithms community interest recently. In particular, a lot of attention has been drawn to the heavy-tailed, α-stable distribution [4,13,14,17,21,22]. It turns out that evolutionary algorithms which make use of distribution from this class gain abilities that allow them to find a balanced compromise between exploitation and exploration of the search space [16].

Till now, the application of the multidimensional stable distributions to global optimization algorithms has been limited to the simplest cases: the mutation of the base point has obtained by adding a random vector composed of stable, independent, random variables [13,14,21,22], or a isotropic random vector [15,17]. This limitation causes that many properties of the stable distributions, which can turn out valuable in the context of optimization processes, are not exploited. In order to obtain the possibility of modeling of the complicated dependence between decision variables, the Discrete Spectral Measure (DSM) is used to generate a wide class of random vectors in this paper.

The paper is organized as follows. Multivariate α-stable distributions are defined in section 2. Next section contains description of definition and main properties of stable random vectors based on the discrete spectral measure. The advantages of such multivariate random vectors application to mutation operator is illustrated in section 4. The last section summarizes considerations.

2 Multivariate α-Stable Distribution

Let us start this section with a brief introduction of the concept of multivariate α-stable distributions. Generally, a stable multivariate distribution can be defined as follows.

Definition 1. *The random vector $\boldsymbol{X} = [X_1, X_2, \ldots, X_n]^T$ is stable in $\mathbb{R}^n$ if, and only if*

$$\forall A, B > 0 \quad \exists C > 0 \quad \exists \boldsymbol{D} \in \mathbb{R}^n : \quad A\boldsymbol{X}^{(1)} + B\boldsymbol{X}^{(2)} \overset{d}{=} C\boldsymbol{X} + \boldsymbol{D}, \tag{1}$$

where $\boldsymbol{X}^{(1)}$,$\boldsymbol{X}^{(2)}$ are independent copies of the random vector $\boldsymbol{X}$ and $\overset{d}{=}$ means that the left and right random vectors have the same distribution.

Above definition emphasis the most characteristic property of the stable random vectors. The necessary and sufficient conditions of the random vector stability are included in the form of its characteristic function.

Definition 2. *The random vector $\boldsymbol{X}$ is stable if, and only if its characteristic function*
$\varphi(\boldsymbol{k}) = E[\exp(-i\boldsymbol{k}^T\boldsymbol{X})]$ has the following form:

$$\varphi(\boldsymbol{k}) = \exp\left(-\int_{S^{(d)}} |\boldsymbol{k}^T\boldsymbol{s}|^\alpha \left(1 - i\mathrm{sign}(\boldsymbol{k}^T\boldsymbol{s}) \tan\left(\frac{\pi\alpha}{2}\right)\right) \mathit{\Gamma}(d\boldsymbol{s}) + i\boldsymbol{k}^T\boldsymbol{\mu}\right) \tag{2}$$

for $\alpha \neq 1$, and

$$\varphi(\boldsymbol{k}) = \exp\left(-\int_{S^{(d)}} |\boldsymbol{k}^T\boldsymbol{s}| \left(1 - i\frac{2}{\pi}\mathrm{sign}(\boldsymbol{k}^T\boldsymbol{s})\ln|\boldsymbol{k}^T\boldsymbol{s}|\right)\Gamma(d\boldsymbol{s}) + i\boldsymbol{k}^T\boldsymbol{\mu}\right) \tag{3}$$

for $\alpha = 1$, *where* $\Gamma(\cdot)$ *is the so-called spectral measure, and* $\boldsymbol{\mu}$ *stands for shift vector.*

It turns out that a pair $\{\Gamma, \boldsymbol{\mu}\}$ uniquely determine stable distribution [18]. It is worth to notice that any linear combination of components of the stable vector described by definition 2 is univariate α-stable variable $S_\alpha(\sigma, \beta, \mu)$ [23], where $\alpha \in (0, 2]$ is the so-called stability index, $\sigma \in \mathbb{R}_+$ stands for the scale parameter, $\beta \in [-1, 1]$ is skewness coefficient and $\mu \in \mathbb{R}$ is a shift parameter. It must to stressed, that the definition of stable vectors is not straightforward and the presence of spectral measure Γ causes that the class is not an ordinary parametric family. In consequence, a direct definition in practical applications is used rather occasionally. Indeed, in the subsequent part of the paper, our attention is restricted only to the class of stable distributions with discrete spectral measure which possess decidedly simpler form.

3 Stable Distributions with Discrete Spectral Measure

A DSM Γ can be defined by means of Delta Dirac distribution in the following way:

$$\Gamma(\cdot; \boldsymbol{\xi}, \boldsymbol{\gamma}) = \sum_{i=1}^{n_s} \gamma_i \delta_{\boldsymbol{s}_i}(\cdot), \tag{4}$$

where $\boldsymbol{\xi} = \{\boldsymbol{s_i}\}_{i=1}^{n_s}, \boldsymbol{s_i} \in \partial S^{(d)}$ is a set of support points concentrated on a surface of a d-dimensional unit sphere, and $\boldsymbol{\gamma} = \{\gamma_i\}_{i=1}^{n_s}, \gamma_i \in \mathbb{R}_+$ stands for the set of their weights. In this way, for every set $A \subset \partial S^{(d)}$ its measure is given by:

$$\Gamma(A) = \sum_{i=1}^{n_s} \gamma_i I_A(\boldsymbol{s}_i), \tag{5}$$

where $I_A(\cdot)$ is an indictor function of the set A. Characteristic functions (2) and (3) in the case of spectral measure (4) has the form [12]:

$$\varphi(\boldsymbol{k}) = \exp\left(-\sum_{i=1}^{n_s} \gamma_i |\boldsymbol{k}^T\boldsymbol{s_i}|^\alpha \left(1 - i\mathrm{sign}(\boldsymbol{k}^T\boldsymbol{s_i})\tan\left(\frac{\pi\alpha}{2}\right)\right) + i\boldsymbol{k}^T\boldsymbol{\mu}\right) \tag{6}$$

for $\alpha \neq 1$, and

$$\varphi(\boldsymbol{k}) = \exp\left(-\sum_{i=1}^{n_s} \gamma_i |\boldsymbol{k}^T\boldsymbol{s_i}| \left(1 - i\frac{2}{\pi}\mathrm{sign}(\boldsymbol{k}^T\boldsymbol{s_i})\ln|\boldsymbol{k}^T\boldsymbol{s_i}|\right) + i\boldsymbol{k}^T\boldsymbol{\mu}\right) \tag{7}$$

for $\alpha = 1$.

The definition of the DSM allows to use multivariate stable distributions in the simpler way. It is worth to notice that application of the DSM does not limit any properties of multivariate stable vectors. It can be proved the following theorem [2].

Theorem 1. *Let $p(\boldsymbol{x})$ be a density function of the stable distribution described by the characteristic function* (2) *and* (3)*, and $p^*(\boldsymbol{x})$ is a density function of the random vector described by the characteristic function* (6) *and* (7)*. Than*

$$\forall \varepsilon > 0 \quad \exists n_s \in \mathbb{N} \quad \exists \boldsymbol{\xi}, \boldsymbol{\gamma} \quad \forall \boldsymbol{x} \in \mathbb{R}^d : \quad \sup_{\boldsymbol{x} \in \mathbb{R}^d} |p(\boldsymbol{x}) - p^*(\boldsymbol{x})| < \varepsilon. \tag{8}$$

In other words, each stable distribution can be approximated by some distribution based on the DSM with any accuracy. Especially, the existence of a procedure of pseudo-random vectors generation is very important. It turns out, that a simulation procedure of stable random vectors $\boldsymbol{X}$ defined by characteristic functions (6) and (7) is straightforward, and can be implemented making use of the following stochastic decomposition:

$$\boldsymbol{X} \overset{d}{=} \begin{cases} \sum_{i=1}^{n_s} \gamma_i^{1/\alpha} Z_i \boldsymbol{s}_i & \text{for } \alpha \neq 1, \\ \sum_{i=1}^{n_s} \gamma_i^{1/\alpha} \left(Z_i - \frac{2}{\pi} \ln(\gamma_i) \right) \boldsymbol{s}_i & \text{for } \alpha = 1, \end{cases} \tag{9}$$

where Z_i are i.i.d. stable random variables $S_\alpha(1,1,0)$ for which an effective generator can be found in [18].

Random vectors ($\boldsymbol{X} = [X_1, X_2, \ldots, X_n], X_i \sim S_\alpha S(\sigma)$) composed of independent symmetric elements possess a special status in application to mutation operators of EAs in the literature [9,13,14,15,21,22]. It occurs that random vectors can be enriched by addition μ and β parameters, i.e. $\boldsymbol{X} = [X_1, X_2, \ldots, X_n], X_i \sim S_\alpha(\sigma, \beta, \mu)$ if the DSMs are applied. It means that each component acquires additional degrees of freedom. This fact is very important in the context of application of the above random vector to modeling complicated dependencies between decision variables. It is easy to show that exploration such dependencies and their including to a mutation operator accelerates the optimization process. In order to illustrate the possibilities the DSM representation of the random vectors, it can be mention that the vector with independent components $X_i \sim S_\alpha S(\sigma)$ have the DSM focused in the points of orthogonal axes and surface of the unit sphere intersection with a different weights. The versatility of the DSM representation of the distribution is included in [18]

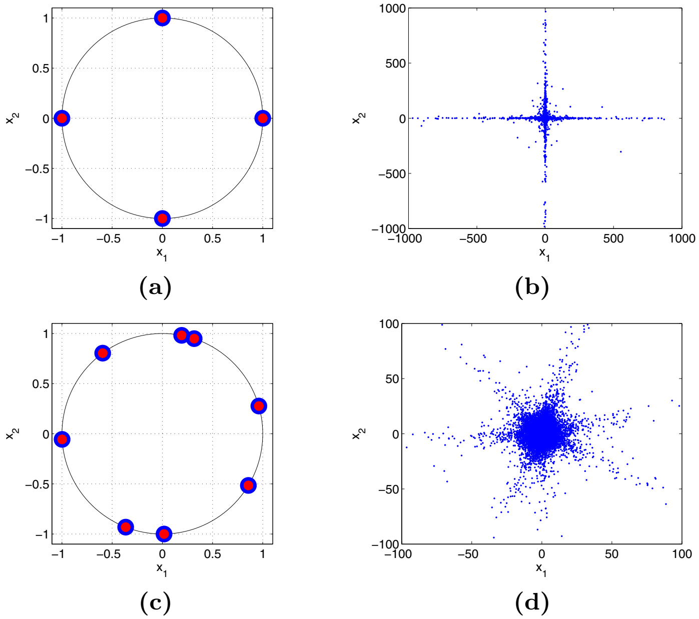
Theorem 2. *The spectral measure of the stable vector $\boldsymbol{X}$ is described by a finite number of the support vectors $\boldsymbol{s}_i$ if, and only if the vector $\boldsymbol{X}$ can be represented by a linear combination of the independent stable random variables, i.e:*

$$\boldsymbol{X} = \boldsymbol{A}\boldsymbol{Z}, \tag{10}$$

where $\boldsymbol{A} \in \mathbb{R}^{d \times N}$, $\boldsymbol{Z} = [Z_1, \ldots, Z_N]^T, Z_i \sim S_\alpha(\sigma, \beta, \mu)$.

Basing on the theorem 2, it can be shown that the DSM can be also applied to represent vectors which are described by parameters σ, β, μ and by stochastic dependencies between these vectors.

One of the important properties of the stable distribution based on the DSM and a finite set of support vectors is that almost all probability mass remote from the base point is focused around directions described by support vectors.



(a) (b)

(c) (d)

Fig. 1. Distribution of the DSM support vectors and corresponded random realizations : **(a)**,**(b)** - $\alpha = 0.75$, **(c)**,**(d)** - $\alpha = 1.5$

So, macromutations are took place only in direction parallel to DSM support vectors. This effect is illustrated in Fig. 1 Summarizing, two main benefits obtained by application the DSM representation of multivariate stable distribution can be distinguished: macromutations which allows to simple cross saddles of the searching environments, and possibility of modeling of complex stochastic dependencies. Above benefits are illustrated by the sample described in the next section.

4 Choice of the Optimal Stable Distribution - Computation Sample

Let the 2-dimensional Rastringin's function

$$\phi(\boldsymbol{x}) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2 \qquad (11)$$

be chosen as a objective function for considered computation sample. Moreover, we assume that the point $\boldsymbol{x}_k = [0, -2]^T$ ($\phi(\boldsymbol{x}_k) = 401$) is the current approximation of the optimum. The our goal is the correction of the current solution by a perturbation using a stable random vector $\boldsymbol{X}_{\boldsymbol{\xi}}^{\gamma}$. The aim of this calculation is the selection the optimal stable model from the class of multivariate distributions described by the DSM. First, let us reduce the set of rival probabilistic models to a set of four stable distributions $\Omega = \{\boldsymbol{X}_{\boldsymbol{\xi}}^{\gamma}(\alpha) | \alpha = 0.5, 1.0, 1.5, 2.0\}$.

Each random vector $\boldsymbol{X}_{\boldsymbol{\xi}}^{\boldsymbol{\gamma}}(\alpha)$ is described by the DSM spread on 16 uniformly distributed support points:

$$\boldsymbol{\xi} = \Big\{ \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right], \left[\begin{smallmatrix}0.92\\0.38\end{smallmatrix}\right], \left[\begin{smallmatrix}0.7\\0.7\end{smallmatrix}\right], \left[\begin{smallmatrix}0.38\\0.92\end{smallmatrix}\right], \left[\begin{smallmatrix}0\\1\end{smallmatrix}\right], \left[\begin{smallmatrix}-0.38\\0.92\end{smallmatrix}\right], \left[\begin{smallmatrix}-0.7\\0.7\end{smallmatrix}\right], \left[\begin{smallmatrix}-0.92\\0.38\end{smallmatrix}\right], \cdots$$

$$\cdots, \left[\begin{smallmatrix}-1\\0\end{smallmatrix}\right], \left[\begin{smallmatrix}-0.92\\-0.38\end{smallmatrix}\right], \left[\begin{smallmatrix}-0.92\\0.38\end{smallmatrix}\right], \left[\begin{smallmatrix}-1\\0\end{smallmatrix}\right], \left[\begin{smallmatrix}-0.92\\-0.38\end{smallmatrix}\right] \Big\}.$$

The criterion of the best model selection from the set Ω is chosen in the form:

$$\boldsymbol{\gamma}^* = \arg \min_{\boldsymbol{\gamma} \in \mathbb{R}_+^{16}} C(\boldsymbol{\gamma}), \tag{12}$$

where

$$C(\boldsymbol{\gamma}) = E\left[\min\left\{\frac{\phi(\boldsymbol{x}_k + \boldsymbol{X}_{\boldsymbol{\xi}}^{\boldsymbol{\gamma}}(\alpha))}{\phi(\boldsymbol{x}_k)}, 1\right\}\right]. \tag{13}$$

It occurs that the function (13) does not possess an analytical form, thus, the problem (12) cannot be solved using standard optimization techniques. One of the possible solutions is the Monte Carlo method application [7,10]. Law of large numbers [3] allows to approximate the expectation value (13) using the following estimator:

$$\hat{C}(\boldsymbol{\gamma}) = \frac{1}{N}\sum_{i=1}^{N} \min\left\{\frac{\phi(\boldsymbol{x}_k + \boldsymbol{X}_{i,\boldsymbol{\xi}}^{\boldsymbol{\gamma}}(\alpha))}{\phi(\boldsymbol{x}_k)}, 1\right\}, \tag{14}$$

where $\{\boldsymbol{X}_{i,\boldsymbol{\xi}}^{\boldsymbol{\gamma}}(\alpha))\}_{i=1}^{N}$ stands for a sequence of independent realizations of the random vector with the α-stable distribution. Using the estimator (14), the problem (12) can rewritten into the form:

$$\boldsymbol{\gamma}^* = \arg \min_{\boldsymbol{\gamma} \in \mathbb{R}^{16}} \hat{C}(\boldsymbol{\gamma}). \tag{15}$$

The important problem of such transformation of the optimization problems is a suitable selection of the number N of samples. It is worth to stress that the objective function possess a stochastic property, and the number N controls the variance expectation (14). The large value of N seems to be the best solution. But, on another hand, this solutions leads to a great calculation effort. In order to rational controlling of the estimator (14) quality, the Chernoff's inequality can be applied [20].

Because the objective function possesses a stochastic properties and in order to achieve the compromise between the computation complexity and the estimator quality, the SPSA algorithm [19] is chosen for solving considered problem. This algorithm is dedicated to optimization of the multivariate stochastic functions. Obtained results are presented in Tab 1.

Results of the experiment presents the advantage of distributions with low stability index α. It is surprising, they are more efficient for the unimodal Rastringin's function then the normal distribution ($\alpha = 2$), in spite of their tendency to macromutations. The main cause of this fact is spherical symmetry of the normal distribution. This property of normal distribution causes the fact, that the

Table 1. Objective function values and, corresponding to them, probabilities of success obtained for pseudo-random stable distributions

α	2.0	1.5	1.0	0.5
$C(\gamma^*)$	0.6683	0.4978	0.6522	0.5385
P_s	0.4852	0.6445	0.5145	0.6638

probability of success can not be higher the 0.5. In the case of $\alpha < 2$, when the distribution are non-isotropic, the probability of generation better solution then the base point $\boldsymbol{x}_k$ is over 0.6.

5 Conclussion

Till now, the application of the multidimensional stable distributions to global optimization algorithms has been limited to the simplest cases: the mutation of the base point has obtained by adding a random vector composed of stable, independent, random variables, or a isotropic random vector. This limitation causes that many properties of the stable distributions, which can turn out valuable in the context of optimization processes, are not exploited. In order to obtain the possibility of modeling of the complicated dependence between decision variables, the discrete spectral measure is used to generate a wide class of random vectors in this paper. The main properties of the multivariate stable random vectors based on the DSM are presented in the work and a simple experiment which illustrates that, for the given objective function, dependencies between random components compliance in mutation operator accelerates the efficiency of the optimum searching.

References

1. Beyer, H.G., Schwefel, H.P.: Evolutionary strategies – a comprehensive introduction. Neural Computing 1(1), 3–52 (2002)
2. Byczkowski, T., Nolan, J.P., Rajput, B.: Approximation of multidimensional stable densities. J. of Mult. Anal. 46, 13–31 (1993)
3. Durrett, R.: Probability: Theory and Examples, 2nd edn. Duxbury Press (1995)
4. Gutowski, M.: Lévy flights as an underlying mechanism for a global optimization algorithm. In: Proc. 5th Conf. Evolutionary Algorithms and Global Optimization, pp. 79–86. Warsaw University of Technology Press (2001)
5. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolutionary strategies. Evolutionary Computation 9(2), 159–195 (2001)
6. Karcz-Dulęba, I.: Asymptotic behaviour of a discrete dynamical system generated by a simple evolutionary process. Int. Journ. Appl. Math. Comput. Sci. 14(1), 79–90 (2004)
7. Kemp, F.: An introduction to sequential Monte Carlo methods. Journal of the Royal Statistical Society D52, 694–695 (2003)
8. Kern, S., Uller, S., Uche, D., Hansen, N., Koumoutsakos, P.: Learning probability distributions in continuous evolutionary algorithms. In: Proc. Workshop on Fundamentals in Evolutionary Algorithms, 13th Int. Colloquium on Automata, Languages and Programming, Eindhoven (2004)

9. Liu, X., Xu, W.: A new filled function applied to global optimization, Comput. Oper. Res. 31, 61–80 (2004)
10. MacKey, D.C.J.: Introduction to Monte Carlo methods. In: Jordan, M.I. (ed.) Learning in Graphical Models. NATO Science Series, pp. 175–204. Kluwer Academmic Press, Dordrecht (1998)
11. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer, London (1996)
12. Nolan, J.P., Panorska, A.K., McCulloch, J.H.: Estimation of stable spectral measures - stable non-Gaussian models in finanse and econometrics. Math. Comput. Modelling 34(9), 1113–1122 (2001)
13. Obuchowicz, A.: Evolutionary Algorithms in Global Optimization and Dynamic System Diagnosis. Lubuskie Scientific Society Press, Zielona Góra (2003)
14. Obuchowicz, A., Prętki, P.: Phenotypic Evolution with Mutation Based on Symmetric α-Stable Distributions. Int. J. Applied Mathematics and Computer Science 14, 289–316 (2004)
15. Obuchowicz, A., Prętki, P.: Isotropic Symmetric α-Stable Mutations for Evolutionary Algorithms. In: Proc. IEEE Congress on Evoutionary Computation, CEC 2005, pp. 404–410 (2005)
16. Prętki, P.: α-Stable Distributions in Evolutionary Algorithms of Parametric Global Optimization. PhD Thesis, University of Zielona Góra, Poland (2008) (in Polish)
17. Rudolph, G.: Local convergence rates of simple evolutionary algorithms wich Cauchy mutations. IEEE Trans. Evolutionary Computation 1(4), 249–258 (1997)
18. Samorodnitsky, G., Taqqu, M.S.: Stable Non-Gaussian Random Processes. Chapman & Hall, New York (1994)
19. Spall, J.C.: Introduction to Stochastic Search and optimization. Wiley, Hoboken (1993)
20. Vidysagar, M.: Randomized algorithms for robust controller synthesis using statistical learning theory. Automatica 37, 1515–1528 (2001)
21. Yao, X., Liu, Y.: Fast evolution strategies. In: Angeline, P.J., McDonnell, J.R., Reynolds, R.G., Eberhart, R. (eds.) EP 1997. LNCS, vol. 1213, pp. 151–161. Springer, Heidelberg (1997)
22. Yao, X., Liu, Y., Liu, G.: Evolutionary Programming made faster. IEEE Trans. Evolutionary Computation 3(2), 82–102 (1999)
23. Zolotariev, A.: One-Dimensional Stable Distributions. American Mathematical Society, Providence (1986)

Determining Subunits for Sign Language Recognition by Evolutionary Cluster-Based Segmentation of Time Series

Mariusz Oszust and Marian Wysocki

Rzeszow University of Technology
Department of Computer and Control Engineering
W. Pola 2, 35-959 Rzeszow, Poland
{moszust,mwysocki}@prz-rzeszow.pl

Abstract. The paper considers partitioning time series into subsequences which form homogeneous groups. To determine the cut points an evolutionary optimization procedure based on multicriteria quality assessment of the resulting clusters is applied. The problem is motivated by automatic recognition of signed expressions, based on modeling gestures with subunits, which is similar to modeling speech by means of phonemes. In the paper the problem is formulated, its solution method is proposed and experimentally verified.

Keywords: time series segmentation, multiobjective clustering, evolutionary optimization, sign language recognition.

1 Introduction

Automatic sign language recognition is an important prospective application of gesture-based human-computer interfaces. The aim of the research is a system that properly interprets gestures, e.g. translates them into written or spoken language. Most of such systems described in the literature (see e.g. [1], [2]) are based on word models where one sign represents one model in the model database. They can achieve good performance only with small vocabularies or gesture data sets. The training corpus and the training complexity increase with vocabulary size. So, large-vocabulary systems require the modeling of signed expressions in smaller units than words i.e. the words are modeled with subunits, which is similar to modeling speech by means of phonemes. The main advantage of this approach is that an enlargement of the vocabulary can be achieved by composing new signs through concatenation of subunit models and by tuning the composite models with only small sets of examples. However, an additional knowledge of how to break down signs into subunits is needed.

Different vision-based subunit segmentation algorithms have been developed. Following Liddell and Johnson's movement-hold model the authors of [3] propose modeling each sign (word) as a series of movement and hold segments. Kraiss et al. in [1] present an iterative process of data-driven extraction of subunits using hidden Markov models (HMMs). In all following steps, two state HMMs for

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 189–196, 2010.
© Springer-Verlag Berlin Heidelberg 2010

subunits determined in prior iteration step are concatenated to models of single signs. The boundaries of subunits for the next step result from the alignment of appropriate feature vector sequence to the states by the Vitterbi algorithm. Han et al. in [4] define the subunit boundary using hand motion discontinuity. Temporal clustering by dynamic time warping is adopted to merge similar segments.

In this paper we propose a new approach where the subunits' boundary points are considered as decision variables in a multiobjective optimization problem. The problem consists in finding subunits which can be grouped in clusters of good quality. The quality is measured by two cluster validity indices, one based on entropy [5] and another the Dunn's index [6], [7]. The indices are optimized simultaneously using lexicographic ordering [8] and an immune-based evolutionary algorithm [9], [10]. The approach refers to clustering of time series data [11], [12], multiobjective clustering [13], [14], and cluster-based time series segmentation [15]. The contribution of the paper lies in (1) formulation of the problem of determining subunits for sign language recognition as a multiobjective cluster optimization, (2) proposition of a solution method, and (3) verification of the approach by experiments on both synthetic and real data.

The rest of the paper is organized as follows. Section 2 contains formulation of the problem. Section 3 gives the details of the proposed solution method. The results of experiments using synthetic data, as well as data obtained for isolated words of the Polish Sign Language (PSL) are given in section 4. Section 5 concludes the paper.

2 Problem Formulation

Let $S = \{X_1, X_2, \ldots, X_n\}$ denote a data set, where $X_i = \{x_i(1), x_i(2), \ldots, x_i(T_i)\}$ is a sequence of real valued vectors representing a signed word. All feature vectors $x_i(t), t\epsilon\{1, 2, \ldots, T_i\}, i\epsilon I = \{1, 2, \ldots, n\}$ have identical structures. The integers $t = 1, 2, \ldots$ represent equidistant time points. Two time sequences X_i and $X_{j\neq i}$ may represent different words or different realizations of the same word.

Let us consider a decomposition D, which, for each $i\epsilon I$, defines a number $k_i = k_i(D) \geq 1$ and k_{i-1} cut points $t_i^j = t_i^j(D)$, where $1 < t_i^1 < t_i^2 < \ldots < t_i^{k_i-1} < T_i$. The decomposition means that X_i is partitioned into k_i subsequences. The first subsequence $s_i^1(D)$ starts at $t = 1$ and ends at $t = t_i^1$, the next subsequence $s_i^2(D)$ starts at $t = t_i^1$ and ends at $t = t_i^2$, and so on until the last subsequence $s_i^{ki}(D)$ which starts at $t = t_i^{k_i-1}$ and ends at T_i. The resulting data set $S'(D) = \{s_1^1(D), \ldots, s_1^{k_1(D)}(D), s_2^1(D), \ldots, s_2^{k_2(D)}(D), \ldots, s_n^1(D), \ldots, s_n^{k_n(D)}(D)\} = \{s'_1, s'_2, \ldots, s'_{n'}\}$ contains $n' = n'(D) = \sum_{i=1}^{n} k_i(D)$ sequences. The length of each subsequence is constrained by the minimal l_{min} and the maximal l_{max} number of points. We propose determining a good decomposition into subsequences by solving a multicriteria decision problem, based on the following main steps: (i) partition the set $S'(D)$ into m (a given number) clusters, i.e. $S'(D) = \{C_1(D), C_2(D), \ldots, C_m(D)\}$, (ii) evaluation of the decomposition D

using a vector of $p > 1$ criteria (indices) $J(D) = [J_1(D), J_2(D), \ldots, J_p(D)]$ which characterizes the quality of the resulting clusters. In next sections we suggest a solution method and we show the results of experiments on both synthetic and real data sequences.

3 Basic Elements of the Solution Method

3.1 Distance Measure

To compare discrete sequences we use dynamic time warping (DTW) [6], [16]. Given two time series $Q = \{q(1), q(2), \ldots, q(T_q)\}$ and $R = \{r(1), r(2), \ldots, r(T_r)\}$ DTW aligns the two series so that their difference is minimized. To this end, a $T_q \times T_r$ matrix, where the (i, j) element of the matrix contains the distance $d(q(i), r(j))$ between two points $q(i)$, and $r(j)$. Usually the Euclidean distance is used. A warping path, $W = w_1, w_2, \ldots, w_K$ where $max(T_q, T_r) \leq K \leq T_q + T_r - 1$, is a set of matrix elements that satisfies three constraints: boundary condition, continuity and monotonicity. The boundary condition constraint requires the warping path to start and finish in diagonally opposite corner cells of the matrix. That is $w_1 = (1, 1), w_K = (T_q, T_r)$. The continuity constraint restricts the allowable steps to adjacent cells. The monotonicity constraint forces the points in the warping path to be monotonically arranged in time. The warping path that has the minimum distance $d_{DTW} = \sum_{k=1}^{K} \frac{w_k}{K}$ between the two series is of interest. Dynamic programming is used to effectively find this path. To prevent pathological warping, where a relatively small section of one sequence maps to a much larger section of another, warping window constraints are applied which, additionally, speed up the computation [16]. The warping window usually defines the search region as a narrow strip around the diagonal connecting points w_1, w_K.

3.2 Clustering Procedure

As the clustering algorithm we propose minimum entropy clustering (MEC) described in [5]. Entropy is a measure of information and the uncertainty of a random variable. The method uses entropy measured on a posteriori probabilities as the criterion of clustering. In fact, it is the conditional entropy of clusters given the observations. The problem of clustering consists of two subproblems (1) estimating a posteriori probabilities and (2) minimizing the entropy. Experiments presented in [5] show that MEC performs significantly better than k-means clustering, hierarchical clustering, SOM and EM. Moreover, it can correctly reveal the structure of data and effectively identify outliers simultaneously.

In our problem we used the Java package prepared by the authors of [5] and accessible online [17]. As it performs clustering of vector defined data we considered two approaches based on $n'(D)$ similarity vectors representing the set $S'(D)$ of subsequences to be clustered. Each of the similarity vectors has $n'(D)$ elements where the $j - th$ element of the $i - th$ similarity vector is determined as the DTW distance between the subsequences s'_i and s'_j in the set $S'(D)$. In the first case MEC performs clustering of the similarity vectors. Alternatively, shorter vectors obtained from the similarity vectors by the PCA can be used.

3.3 Clustering Results Evaluation

The vector index $J(D)$ introduced in section 2 actually contains two elements. The first, more important, is the conditional entropy minimized by MEC. The second in the hierarchy is the Dunn's index DI [6], [7]. It is defined by two parameters: the diameter $diam(C_i)$ of the cluster C_i and the set distance $\delta(C_i, C_j)$ between C_i and C_j, where

$$diamC_i = max_{x,y \epsilon C_i}\{d(x,y)\}, \delta(C_i, C_j) = min_{x \epsilon C_i, y \epsilon C_j}\{d(x,y)\} \tag{1}$$

and $d(x,y)$ indicates the distance between points x, y.

$$DI = min_{1 \leq j \leq m}\{min_{1 \leq i \leq m, i \neq j}\{\frac{\delta(C_i, C_j)}{max_{1 \leq k \leq m} diamC_k}\}\} \tag{2}$$

Larger values of DI correspond to good clusters.

Note that the distances needed in DI can be considered as distances between the similarity vectors or, alternatively, as distances between respective sequences. Obviously, in the second approach necessary information is extracted from the similarity vectors.

3.4 Optimization Algorithm

As follows from section 3.3 our problem is a multiobjective optimization problem (MOP) with two criteria. To solve MOPs evolutionary algorithms are often used. Evolutionary algorithms deal simultaneously with a set of possible solutions (the so called population) which allow us to find several members of the Pareto optimal set in single run of the algorithm [18].

Our approach to solve the MOP adopts the immune-based algorithm CLONALG originally used for single-objective optimization [9], [10]. We use lexicographic ordering [8]. Here the single objective J_1 (considered the most important) is optimized without considering J_2. Then the J_2 is optimized but without decreasing the quality of the solution obtained for J_1. In the sequel we shortly describe the algorithm, the encoding method, and the mutation operator.

CLONALG. The main loop (repeated *gen* times, where *gen* is the number of generations) consists of four main steps: one initial step where all the elements of the population are evaluated and three transformation steps: clonal selection, mutation, apoptosis.

1. Evaluation. For each element D in the population P compute $J_i(D), i = 1, 2$ and perform lexicographic ordering of the elements.
2. Clonal selection. Choose a reference set $P_a \subset P$ consisting of h elements at the top of the ranking obtained in step 1.
3. Mutation.

3.1. For each $D \epsilon P_a$ make c mutated clones $Dc_j, j = 1, 2, \ldots, c$, compute their values $J_1(Dc_j), J_2(Dc_j)$, and place the clones in the clonal pool CP.
3.2. Lexicographically order the elements of $P \cup CP$, choose a subset $P_c \subset P \cup CP$ containing N best elements, where N denotes the size of P.

4. Apoptosis. Replace b worst elements in P_c by randomly generated elements.
5. Set $P \subset PC$.

In the algorithm the current population P is mixed with the clonal pool CP and the predefined number of best elements (i.e. at the top of the ranking) is picked up to form new population. The last step of the main loop replaces b worst solutions by randomly generated elements.

Encoding and mutation. Each element of the population P represents a decomposition D of the set S into a set S'(see section 2). It has the form of the integer valued vector $D = [t_1^1, t_1^2, \ldots, t_1^{k_1-1}, t_2^1, t_2^2, \ldots, t_2^{k_2-1}, \ldots, t_n^1, t_n^2, \ldots, t_n^{k_n-1}]$ composed of the cut points of the original sequences. The mutation process consists of a given number M of mutations conducted on a population element. The mutation means an operation randomly chosen from the following variants: (a) add cut point (probability 1/4), (b) remove cut point (probability 1/4), (c) move cut point (probability 1/2). In all cases a subsequence is randomly selected and, depending on a drown variant, it is: (a) divided into two shorter subsequences, (b) joined together with its preceding subsequence, (c) made shorter or longer by shifting its initial point. The new cut point in (a) and (c) is placed in a position randomly chosen from the corresponding set of feasible points, i.e. the points for which the resulting subsequences satisfy the length constraints. Similarly, the union in (b) is accepted if the resulting subsequence is not too long.

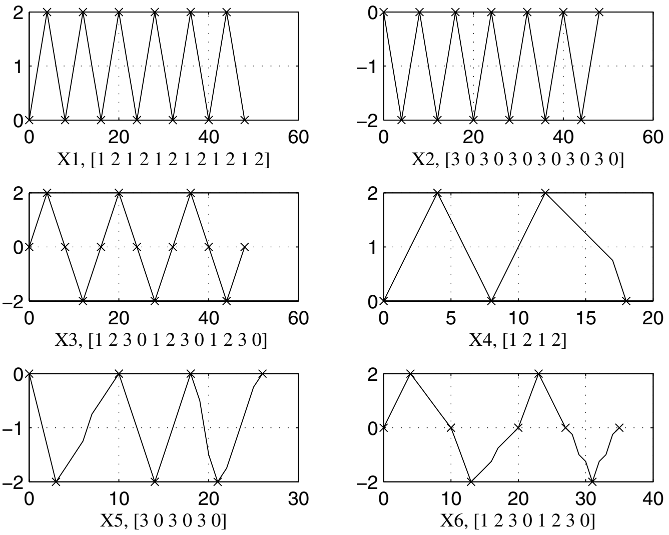
4 Experiments

In this section we present results of two experiments. In the first case synthetic data are considered, the other experiment is based on real sequences obtained for signed Polish words.

4.1 Synthetic Data

The set S consists of six sequences presented in fig.1. In each sequence one can distinguish subsequences which are identical or mutually related by a nonlinear time scale transform. We considered partitioning of the sequences into subsequences which can be grouped into (i) two clusters ($m = 2$), (ii) four clusters ($m = 4$). In both cases the minimum (l_{min}) and the maximum (l_{max}) subsequence lengths (see section 2) are defined.

The following parameters of the optimization procedure were used in the experiment: $N = h = 100, c = 15, b = 10, M = 2, gen = 60, l_{min} = 6, l_{max} = 12$ (2 clusters), $l_{min} = 4, l_{max} = 8$ (4 clusters). The best result obtained for $m = 4$ is characterized in fig. 1. Automatically obtained partitioning for $m = 2$ was also consistent with the result expected by human.


X1, [1 2 1 2 1 2 1 2 1 2 1 2]

X2, [3 0 3 0 3 0 3 0 3 0 3 0]

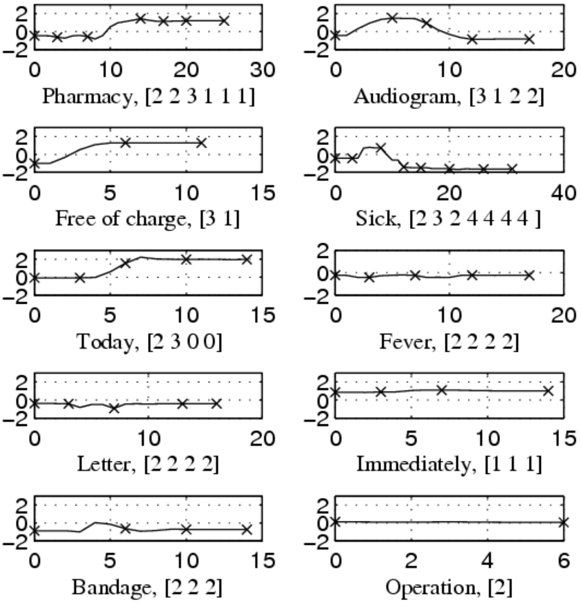
X3, [1 2 3 0 1 2 3 0 1 2 3 0]

X4, [1 2 1 2]

X5, [3 0 3 0 3 0]

X6, [1 2 3 0 1 2 3 0]

Fig. 1. Sequences $X_1 - X_6$ used in the experiment; automatically determined subsequences' boundaries for $m = 4$ are marked, resulting transcriptions based on four subunits $0, \ldots, 3$ are given in brackets


Pharmacy, [2 2 3 1 1 1]

Audiogram, [3 1 2 2]

Free of charge, [3 1]

Sick, [2 3 2 4 4 4 4]

Today, [2 3 0 0]

Fever, [2 2 2 2]

Letter, [2 2 2 2]

Immediately, [1 1 1]

Bandage, [2 2 2]

Operation, [2]

Fig. 2. Sequences representing signed words; automatically determined subsequences' boundaries for $m = 5$ are marked, resulting transcriptions based on five subunits $0, \ldots, 4$ are given in brackets

4.2 Real Data

Sequences used in this experiment represent 10 signed words of PSL. Each sequence was chosen as a medoid of 40 realizations of appropriate word performed by two signers. Fig. 2. shows normalized ($mean = 0, stdev = 1$) values of the horizontal placement of the right hand center vs. frame number, obtained from pictures registered by the camera with the rate of 25 f/s. Parameters used in this experiment are the same as in the experiment with four clusters in subsection 4.1. We solved the optimization task for 2, . . . , 6 clusters. The best result (with the greatest value of the Dunn's index) has been obtained for five clusters ($m = 5$). Fig. 2. shows that the subunits with the same labels are quite similar, although of different lengths, whereas the subunits with different labels differ.

5 Conclusions

Large-vocabulary systems of sign language recognition require the modeling of signed expressions in smaller units than words. However, an additional knowledge of how to break down signs into subunits is needed. In vision-based systems the subunits are related to visual information. As linguistic knowledge about the useful partition of signs in regard of sign recognition is not available, the construction of an accordant partition is based on a data-driven process when signs are divided into segments that have no semantic meaning, then similar segments are grouped and labeled as a subunit. In this paper we propose a new approach to determining the subunits. Subunits' boundaries are considered as decision variables in a multiobjective optimization problem. We use two objective functions, entropy and the Dunn's index, as measures of cluster quality. These functions are optimized simultaneously. The method has been successfully verified, but there remain some open questions. The number of clusters is determined experimentally. In the future it will be included as the additional decision variable in the optimization task. Second question concerns including other cluster validity indices and using other optimization approaches. We use lexicographic ordering and an immune-based evolutionary algorithm, but other evolutionary optimization methods may be considered, see e.g. [18]. We will consider these issues in future research. A next step will be related to more advanced experimentation including recognition words and sentences of the PSL.

Acknowledgement. This research was supported by the Polish Ministry of Higher Education under grant N N516 369736.

References

1. Kraiss, K.F.: Advanced Man-Machine Interaction. Springer, Berlin (2006)
2. Ong, S.C.W., Ranganath, S.: Automatic Sign Language Analysis: A Survey and the Future beyond Lexical Meaning. IEEE Trans. PAMI 27, 873–891 (2005)
3. Vogler, C., Metaxas, D.: A Framework for Recognizing the Simultaneous Aspects of American Sign Language. Computer Vision and Image Understanding 81, 358–384 (2001)

4. Han, J., Awad, G., Sutherland, A.: Modelling and Segmenting Subunits for Sign Language Recognition Based on Hand Motion Analysis. Pattern Recognition Letters 30, 623–633 (2009)
5. Li, H., Zhang, K., Jiang, T.: Minimum Entropy Clustering and Applications to Gene Expression Analysis. In: 3rd IEEE Computational Systems Bioinformatics Conference, pp. 142–151 (2004)
6. Xu, R., Wunsch, D.C.: Clustering. J. Wiley Sons, Inc., Hoboken (2009)
7. Maulik, U., Bandyopadhyay, S.: Performance Evaluation of Some Clustering Algorithms and Validity Indices. IEEE Trans. PAMI 24, 1650–1654 (2002)
8. Miettinen, K.M.: Nonlinear Multiobjective Optimization. Kluwer Acad. Publ., Dordrecht (1998)
9. De Castro, L.N., Von Zuben, F.J.: Learning and Optimization Using the Clonal Selection Principle. IEEE Trans. on Evolutionary Computation 6, 239–251 (2002)
10. Trojanowski, K., Wierzchon, S.: Immune-Based Algorithms for Dynamic Optimization. Information Sciences 179, 1495–1515 (2009)
11. Bicego, M., Murino, V., Figueiredo, M.A.T.: Similarity-based Classification of Sequences Using Hidden Markov Models. Pattern Recognition 37, 2281–2291 (2004)
12. Liao, T.W.: Clustering of time series data - a survey. Pattern Recognition 38, 1857–1874 (2005)
13. Handl, J., Knowles, J.: An Evolutionary Approach to Multiobjective Clustering. IEEE Trans. on Evolutionary Computation 11, 56–76 (2007)
14. Saha, S., Bandyopadhyay, S.: A symmetry-based multiobjective clustering technique for automatic evolution of clusters. Pattern Recognition 43, 738–751 (2010)
15. Tseng, V.S., Chen, C.H., Huang, P.C., Hong, T.P.: Cluster-based genetic segmentation of time series with DWT. Pattern Recognition Letters 30, 1190–1197 (2009)
16. Ratanamahatana, C.A., Keogh, E.: Three Myths about Dynamic Time Warping Data Mining. In: SIAM Int. Conf. on Data Mining, pp. 506–510 (2005)
17. Minimum Entropy Clustering Java package, `http://www.cs.ucr.edu/~hli/mec/`
18. Coello Coello, C.A.: Evolutionary Multi-Objective Optimization: A Historical View of the Field. IEEE Computational Intelligence Magazine 1, 28–36 (2006)

Analysis of the Distribution of Individuals in Modified Genetic Algorithms

Krzysztof Pytel[1] and Tadeusz Nawarycz[2]

[1] Academy of Humanities and Economics in Lodz, Poland
[2] Department of Biophysics, Medical University in Lodz, Poland
kpytel@ahe.lodz.pl,
tadeusz.nawarycz@umed.lodz.pl

Abstract. The article presents the results of the analysis of the distribution of individuals in a modified genetic algorithm for solving function optimization problems. In the proposed modification of the genetic algorithm, we use the fuzzy logic controller (FLC). The authors proposed the FLC, which estimates all individuals in the population and modifies the probability of the selection to the parents' pool and the probability of the mutation of their genes. In the article we present the results of the analysis of the distribution of individuals in all generations of the algorithm. We compared the results of the elementary algorithm and the algorithm with the adaptation of the selection and mutation probabilities. The new algorithm has been tested on a number of sophisticated functions with satisfactory results.

Keywords: fuzzy logic, genetic algorithms, artificial intelligence, function optimization.

1 Introduction

Genetic algorithms are a group of evolutional algorithms which imitate the natural processes of evolution in the world of living organisms - i.e. mutation, genetic recombination and natural selection, especially the rule of the "survival of the fittest". The subject of genetic algorithms is considered in [2], [4]. In the process of the evolution GA do not take advantage of the knowledge specific to the task. The information collected in prior generations can help to set a trend for the evolution process. This information allows to point the direction, which can cause improved results of the evolution. The experts' knowledge about the evolution has a descriptive character and is often subjective. It is not possible to express this knowledge with the aid of mathematical rules. For this reason, a fuzzy logic controller (FLC) will set a trend of evolution [6].

The bases of fuzzy logic have been formulated by L. Zadeh. This theory enables the description of a real system with the assistance of the fuzzy notion. The subject of fuzzy logic is considered in [1], [5].

Genetic algorithms, fuzzy logic and neural networks joint in one system let us build a computational intelligence [7], [8]. Such a system is able to fit itself to

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 197–204, 2010.
© Springer-Verlag Berlin Heidelberg 2010

solve problems across the change of parameters of the algorithm so as to obtain the best results.

2 The Principle of the Genetic Algorithm

Genetic algorithms (GA) can process data and information, the same way as biological organisms do. They can solve sophisticated optimization problems, so they are usually used for seeking approximate results of the NP-complete problems. Genetic algorithms operate on a whole population of individuals. Each individual represents a potential solution of the optimization problem and has an associated fitness function to determine, which will be used to produce a new offspring in the reproduction process.

The first generation is usually created randomly. The genetic algorithm seeks for better and better solutions, creating subsequent generations of the individuals in forthcoming iterations (generations). Every individual is evaluated by the fitness function. In every generation a few individuals are selected to the parents' pool (the process of selection). A greater probability of the selection is associated to the individuals having a higher value of the fitness function. The individuals in the parents' pool are associated into couples. They exchange their genetic material, so we receive a new generation of offspring. This part of the algorithm is called the process of crossing-over.

During rewriting of the genetic information in biological systems, there exists the risk of mutation: however, it seldom takes place. In genetic algorithms, fate values will be inserted into some places of an individual's code. A new generation of offspring after the crossing-over and mutation establishes the next generation. The algorithm is stopped when it reaches a satisfying result, when the predetermined number of generations has been created or when the time limit has been reached. The solution is an individual with the highest value of the fitness function, represented by the best chromosome in the final population.

An elementary genetic algorithm consists of the following steps:

1. the choice of the first generation,
2. the estimation of each individual's fitness,
3. the check of the stop conditions,
4. the selection of individuals to the parents' pool,
5. the creation of a new generation with the use of operators of crossing and mutation,
6. the printing of the best solution.

Figure 1 shows the block diagram of the genetic algorithm.

There are two parameters in elementary genetic algorithms which determine the evolution: the probability of the selection to the parents' pool and the probability of the mutation. Genetic algorithms can be improved by the utilization of the knowledge of experts to predict the course of the process of the evolution. The experts' knowledge about the evolution has a descriptive character and is often subjective. That is why we utilize a fuzzy logic controller to set the trend of evolution.

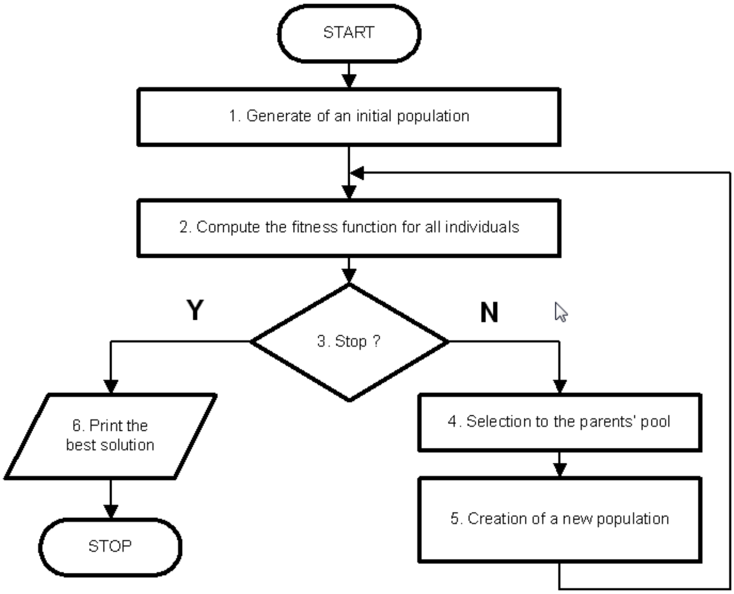

START
1. Generate of an initial population
2. Compute the fitness function for all individuals
Y
N
3. Stop ?
6. Print the best solution
4. Selection to the parents' pool
5. Creation of a new population
STOP

Fig. 1. The block diagram of the genetic algorithm

2.1 Adaptation of the Probability of Selection

The probability of selection determines the ability of an individual to act as a parent for the parents' pool. The chances of the individual for transferring his genetic material to the next generation increase with the probability of selection. Well-adapted individuals are the most wanted ones in the parents' pool: however, weak individuals should also find their way to the parents' pool in order to prevent violent loss of their genetic material and premature convergence. We suggest the modification of the genetic algorithm for the realization of this strategy relying on the introduction of the additional fuzzy logic controller (FLC) for the evaluation of each individual in the population.

The FLC modifies the probability of selection using the following rules:

- enlarge the probability of selection for the individuals with the value of the fitness function of above the average in generations in which the average value of the fitness function grows with relation to the preceding generation,
- don't change the probability of selection for individuals with the value of the fitness function equal to the average in generations in which the average value of the fitness function does not change the relation to the preceding generation,
- diminish the probability of selection for individuals with the value of the fitness function below the average in generations in which the average value of the fitness function decreases with relation to of the preceding generation.

Figure 2. shows the block diagram of the modified genetic algorithm (the block of the fuzzy logic controller is gray).

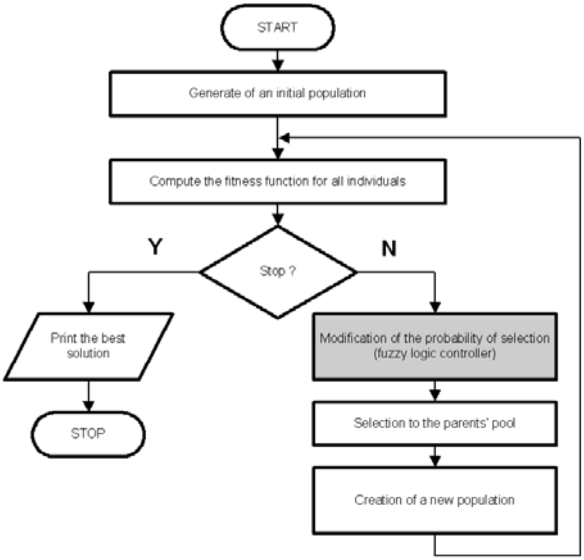

START
Generate of an initial population
Compute the fitness function for all individuals
Y
Stop ?
N
Print the best solution
STOP
Modification of the probability of selection (fuzzy logic controller)
Selection to the parents' pool
Creation of a new population

Fig. 2. The block diagram of the modified genetic algorithm

2.2 Adaptation of the Probability of Mutation

The probability of mutation determines the ability of the algorithm to exploit the search space. In the initial period, mutations are frequent in order to make the search of the whole search space possible (exploration mode). In the final period, mutations are rarer than at the start, which allows the algorithm to concentrate on the searching of the earlier established areas (exploitation mode). The mutation of a gene may cause a new well-adapted individual to shift the algorithm from the local optimum. We suggest the use of the modification of the genetic algorithm for the realization of this strategy which is based on the introduction of the additional FLC for the evaluation of each individual in the population.

The FLC modifies the probability of mutation using the following rules:

- enlarge the probability of mutation of individuals with the value of the fitness function of less then the average in generations in which the average value of the fitness function decreases with relation to the preceding generation,
- don't change the probability of mutation of individuals with the value of the fitness function equal to the average in generations in which the average value of the fitness function does not change in relation to the preceding generation,
- diminish the probability of mutation of individuals with the value of the fitness function above the average in generations in which the average value of the fitness function increases with relation to the preceding generation.

Figure 3. shows the block diagram of the modified genetic algorithm (the block of the fuzzy logic controller is gray). The construction of the fuzzy logic controller in details is considered in [6].

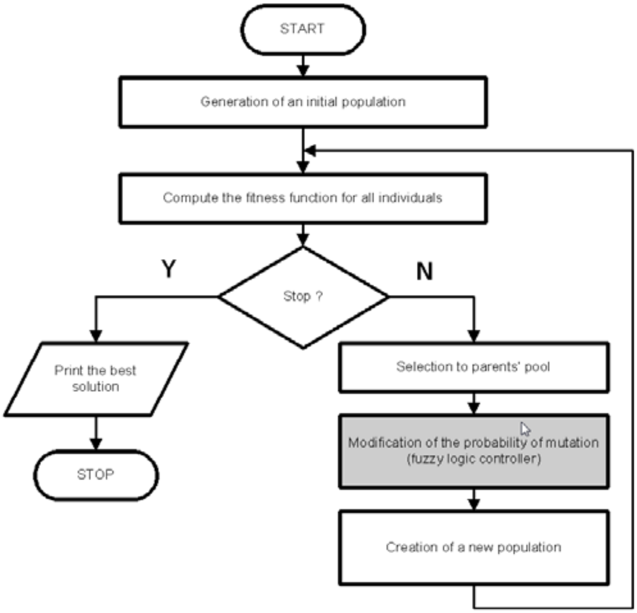

START
Generation of an initial population
Compute the fitness function for all individuals
Y
Stop ?
N
Print the best solution
STOP
Selection to parents' pool
Modification of the probability of mutation (fuzzy logic controller)
Creation of a new population

Fig. 3. The block diagram of the modifed genetic algorithm

3 Computational Experiments

We chose the problem of the optimization of the continuous function for the test. We used the function shown in Figure.4. This function is one of the functions proposed in [3]. It is a sophisticated function with many local optima with different values, which permits to estimate the ability of the algorithm to solve difficult optimization problems. The first generation was placed in the local optimum (point [5, 5]). The algorithm should find the total optimum (point [50, 50]), avoiding the local optima. The algorithm's parameters used in the experiment are:

- the genes of individuals are represented by real numbers,
- the probability of crossover = 0,8,
- the probability of mutation = 0,15,
- the number of individuals in the population = 100,
- the algorithms were stopped after 1000 generations.

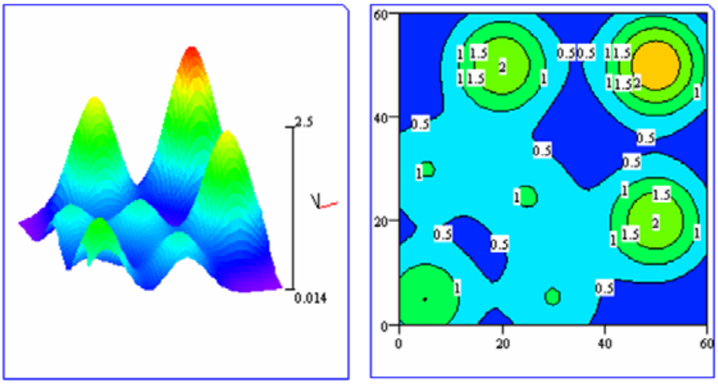


Fig. 4. Surface plot and contour plot of the test function

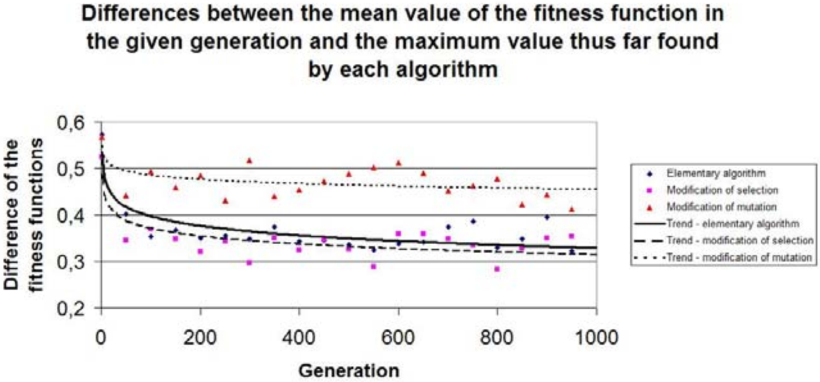


Fig. 5. Differences between the mean value of the fitness function in the given generation and the maximum value thus far found by each algorithm

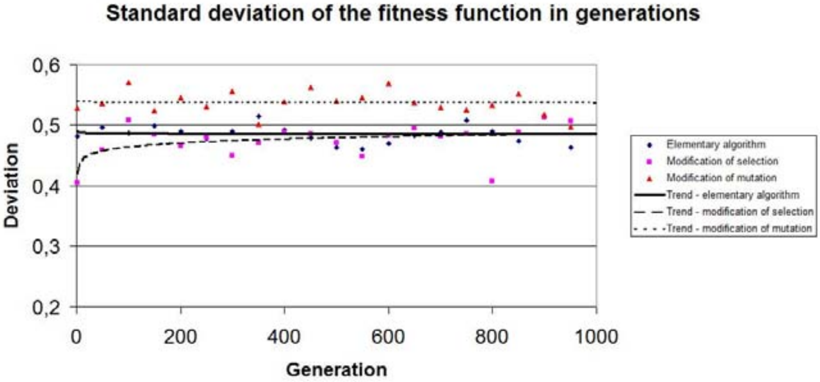


Fig. 6. Changes of the value of the standard deviation of the fitness function of individuals in all the generations of algorithms

The test problem was solved by the elementary algorithm and algorithms with the modification of the probability of selection and mutation. In the experiment we investigated the distribution of the value of the fitness function of individuals in all the generations of the algorithm. All the algorithms were run 10 times. As the factor of the diversity we accepted the difference between the mean value of the fitness function in the given generation and the maximum value thus far found by the algorithm. Additionally, we evaluated the standard deviation of the value of the fitness function of all individuals in every generation.

The graph on Figure 5. illustrates differences between the mean value of the fitness function in the given generation and the maximum value thus far found by each algorithm.

The graph on Figure 6. illustrates changes of the value of the standard deviation of the fitness function of individuals in all the generations of algorithms.

4 Final Remarks

In the algorithm with an adaptation of the probability of the selection the standard deviation of individuals is smaller than in the elementary algorithm. The algorithm found the area quicker, wherein appears the total optimum. The population of individuals is collected in this area. The values of the fitness function of individuals are similar. The difference between the mean value of fitness in the given generation and the maximum value obtained by the algorithm with a modification of the selection is smaller than in the elementary algorithm. The fuzzy adaptation of the probability of selection introduces "harder" methods of selection to the algorithm. The FLC increases the ability of the algorithm to exploit the search space. Better adapted individuals are chosen to the parents' pool, which accelerates the algorithm's convergence. Such a solution can be useful in improving an algorithm's convergence.

In the algorithm with a modification of the probability of mutation the standard deviation of individuals is greater than in the elementary algorithm. The algorithm found the area, wherein appears the total optimum. The population of individuals is collected in this area, but the number of individuals within it is smaller than in the elementary algorithm. The population is more diverse in the genetic sense. The difference between the mean value of the fitness function in the given generation and the maximum value obtained by the algorithm with the modification of mutation is greater than in the elementary algorithm. The FLC enlarges the number of the mutations, holding the greater genetic variety of individuals. The ability to create new, previously non-existent values of genes is alos increased. The fuzzy adaptation improves the abilities of the algorithm to explore the search space. Such a solution can be useful to broaden the searching of all the search space and protect an algorithm from premature convergence to local optima.

References

1. Driankov, D., Hellendoorn, H., Reinfrank, M.: An Introduction to Fuzzy Logic. Springer, Berlin (1993)
2. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Pub. Co., Reading (1989)
3. Kwasnicka, H.: Evolutionary Computation in Artificial Intelligence. In: Publishing House of the Wroclaw University of Technology, Wroclaw, Poland (1999) (in Polish)
4. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer, Berlin (1992)
5. Piegat, A.: Fuzzy modelling and control. Academic Publishing House EXIT, Warsaw (1999) (in Polish)
6. Pytel, K., Kluka, G.: Application of fuzzy logic to aid individual's evolution in genetic algorithms. In: Pytel, K., Kluka, G. (eds.) Studies in Automation and Information Technology. Publishing House of the Poznan Society for the Advancement of the Arts and Sciences, Poznan (2002)
7. Rutkowska, D.: Intelligent Computational Systems. Academic Publishing House PLJ, Warsaw (1997) (in Polish)
8. Rutkowska, D., Pilinski, M., Rutkowski, L.: Neural Networks, Genetic Algorithms and Fuzzy Systems. PWN Scientific Publisher, Warsaw (1997) (in Polish)

Performance Analysis for Genetic Quantum Circuit Synthesis

Cristian Ruican, Mihai Udrescu, Lucian Prodan, and Mircea Vladutiu

Advanced Computing Systems and Architectures Laboratory
University "Politehnica" Timisoara, 2 V. Parvan Blvd., Timisoara 300223, Romania
{crys,mudrescu,lprodan,mvlad}@cs.upt.ro
http://www.acsa.upt.ro

Abstract. Genetic algorithms have proven their ability in detecting optimal or closed-to-optimal solutions to hard combinational problems. However, determining which crossover, mutation or selector operator is best for a specific problem can be cumbersome. The possibilities for enhancing genetic operators are discussed herein, starting with an analysis of their run-time performance. The contribution of this paper consist of analyzing the performance gain from the dynamic adjustment of the genetic operators, with respect to overall performance, as applied for the task of quantum circuit synthesis. We provide experimental results demonstrating the effectiveness of the approach by comparing our results against a traditional GA, using statistical significance measurements.

1 Introduction

The pursuit for performance in computers is relentless. For the so-called classical computers the acquired experience is vast, developed over more than half a century, whereas for quantum computers the race has started relatively recently, in the 1980's. Even from today's perspective, it cannot exactly be foreseen that in the next decade the quantum computer will be physically feasible. Evolutionary search to seek circuit synthesis solutions in a search space was already applied, with the focus being on the analysis of the genetic operators and their performance.

An important view on the optimization problems, the benchmark, is emphasized by the "No Free Lunch" theorem, where it is considered that any elevated performance over one class of problems is exactly paid in terms of performance over another class [7]. The proposed adaptive parameter control algorithm outperforms this limitation -proven by the statistical significance of the experimental results- and highlights the obtained performance by comparison with a conventional GA.

The task of implementing the Meta-Heuristic approach on Quantum Circuit Synthesis (MH-QCS) makes use of ProGA [8] framework, that provides all the necessary support for developing genetic algorithms. Our ProGA framework underpins a robust and optimized environment, its architecture being extended to handle the additional statistical information. The statistical data is processed

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 205–212, 2010.
© Springer-Verlag Berlin Heidelberg 2010

on-the-fly by the adaptive algorithm and the results are used for adjusting genetic operator's rates during run-time. Our proposal focuses on the genetic algorithm parameter control by involving the statistical information, from the current state of the search into algorithm decision. Our experiments prove the fact that a higher convergence rate is reported by genetic evolution and therefore an important runtime speedup is achieved.

The automatic generation of a quantum circuit that will implement a given function is not an easy task; in order to solve this problem the genetic algorithm will evolve a possible solution that will be evaluated against other previous solutions obtained, and eventually a close-to-optimal solution will be indicated. It is hard, if not impossible, to guess the values used for the tuning of genetic algorithm, because even a small change in the circuit topology will generate a different quantum logic function; this is the main motivation for adopting an adaptive genetic algorithm.

2 Background

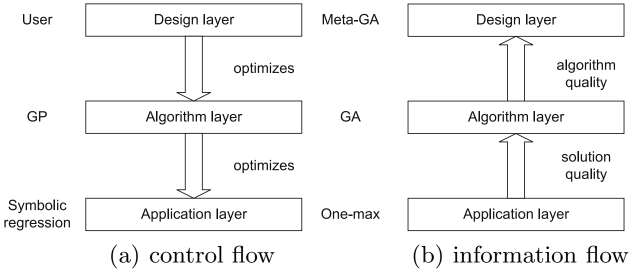
Quantum computation is computation made with coherent atomic scale dynamics. A quantum computer is a physical device able to perform computation driven by quantum mechanical phenomena, such as entanglement and superposition of basis states. For the classical computer, the unit of information is the bit, whereas in quantum computation its counterpart is the so-called qubit. A quantum bit may be represented by using the spin 1/2 particle. For example, a spin-down $|\downarrow\rangle$ and a spin-up $|\uparrow\rangle$ may be used to represent the binary information $|0\rangle$ and $|1\rangle$. In Bra-Ket notation, a qubit is a normalized vector in a two dimensional Hilbert space $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, $|\alpha|^2 + |\beta|^2 = 1$ ($\alpha,\beta \in \mathbb{C}$), where $|0\rangle$ and $|1\rangle$ are the basis states [2].

The Genetic Algorithms (GA) are adaptive heuristic search algorithms based on evolutionary ideas of natural selection used to find solutions for optimization and search problems. The reported results are, in many cases, more efficient, more elegant, and more complex than the solutions discovered by the human mind. The new field of Evolvable Quantum Information (EQI) has been established as the merging of quantum computation and evolvable computation [1].

The problem of setting values for different control parameters is crucial in the context of the algorithm performance. Each GA parameter is responsible with controlling the evolution path towards the solution. There are two major forms of setting the parameter values for a genetic algorithm [6]:

- Parameter tuning: the parameter values are fixed before the algorithm run and remain as such during the algorithm run. There are several disadvantages for the tuning: finding good parameters before the run may be time consuming and it is possible not to get optimal values for all the phases.
- Parameter control: the initial parameter values are changed during the algorithm run, keeping the dynamic spirit of evolution. The adaption algorithm uses the feedback values from the process to adjust the parameters for better performance.

In Figure 1, the upper part of the hierarchy contains a method that aims at finding optimal parameters for the GA, while the lower part is dedicated to possible problem solutions on the application layer. Mainly, the same approach on splitting the design into several layers is also applied within our paper. Thus, the quantum circuit synthesis genetic algorithm will run in the application layer, while the algorithm responsible with the dynamic adjustment of the operators will run in the design layer.



(a) control flow (b) information flow

Fig. 1. The 3-layered hierarchy of parameter tuning [6]

3 Previous Work

In reference [3], Yao was the first one to investigate the effect of the genetic operators, using several genetic algorithms as examples. Three metrics are proposed (e.g. best and average tour length, population diversity) to understand "when and why some operators are useful and how we can combine them together to improve the performance". As conclusion, the study suggests the use of the "greedy" crossover and the "hard" selection with a low mutation rate for better performance. Fredrik [4] considers that different genetic operators may be better suited to exploring the search space at different stages. In [5], Affenzeller has identified that the selection is the "driving force" in GAs. Smit and Eiben [6], consider the tuning for the algorithm parameters as a non-trivial problem, which is essential for a good algorithm performance. It is expressed that is difficult to choose between a good crossover operator (defined in a finite domain) and a good crossover rate p_c (defined in a subset of $\mathbb{R}$). The difference is essential because for the given rates it is possible to apply heuristic search and optimization techniques to find optimal values, while for the operators only the enumeration values are possible.

4 Genetic Algorithm Details

A successful use of meta-heuristics to difficult optimization problems (like quantum circuit synthesis) usually depends on problem specific encoding. Following Nature, where a chromosome is composed of genes, within our chromosome the genes correspond to the circuit sections, thus allowing to represent any possible solution for circuit synthesis (see Figure 2 a). Any gene will store the specific characteristics of a particular section, and the genetic operators will be applied at the gene level or inside the gene. The terminal set is composed of quantum gates (randomly used in the chromosome encoding) and the function set is composed of the mathematical functions necessary to evaluate the circuit output function (tensor product, multiplication and equality). The fitness measure specifies what the user expects from the synthesis algorithm. Therefore, the fitness assignments to a chromosome indicate how close the individual output is to the algorithm target. Considering the discrete search space $\mathbb{X}$ and the objective function $f : \mathbb{X} \rightarrow \mathbb{R}$, our scope is the find $max_{x \in \mathbb{X}} f$ where x is a vector of decision variables, $f(x) = f(x_1, ..., x_n)$. It is a maximization problem, because the goal is to find the optimum quantum circuit that implements a given input function. The fitness function is defined as:

$$eval(x) = f(x) + W \times penalty(x) \tag{1}$$

where

$$f = \frac{f(evolved\ circuit)}{f(initial\ circuit)} \tag{2}$$

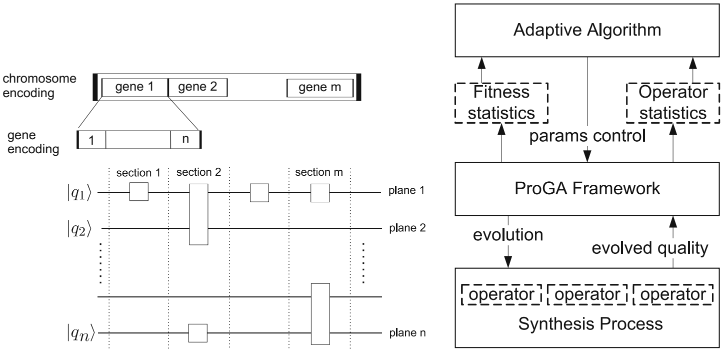
and

$$penalty = 1 - \frac{number\ of\ evolved\ gates - number\ of\ initial\ gates}{number\ of\ initial\ gates} \tag{3}$$

From a meta-heuristic point of view, it is considered that genetic algorithms contain all necessary information for adaptive behavior. Two types of statistical data are used as input for the adaptive algorithm (see Figure 2 b). The first type is represented by the fitness results for each population corresponding to the best, mean and worst chromosomes. The second type is represented by the operator performance. Following an idea proposed by Gheorghies[12], the performance records are essential in order to decide on operators reward ("absolute" when the resulted offspring has a higher fitness than the best fitness from the previous generation; "relative" when the resulted offspring has a better fitness than its parents, but it is not absolute; "in range" when the resulted offspring has a fitness situated between the fitness values of its parents; and "worse" when the resulted offspring has a fitness value that is lower than that of its parents).

From the adaptive algorithm it is not essential to know operator implementation details; instead, one has to be informed about the number of operators because, for each of them, a separate statistical structure will be reserved. The adaptive algorithm will receive breeding feedback from each operator and will analyze the data, in order to compute the operator performance and decide on its adjustment rate. The adaptive algorithm distinguishes between the quality of solutions evolved by different operators and adjusts the rates, based on merits.

An important concept throughout the rest of the paper is related to the efficiency of controlling the algorithm parameters; it implies additional computation power. We will illustrate the feasibility of the control algorithms and their payoff in terms of performance.



(a) encoding (b) control

Fig. 2. Chromosome Encoding (a) and Adaptive Control Layers (b)

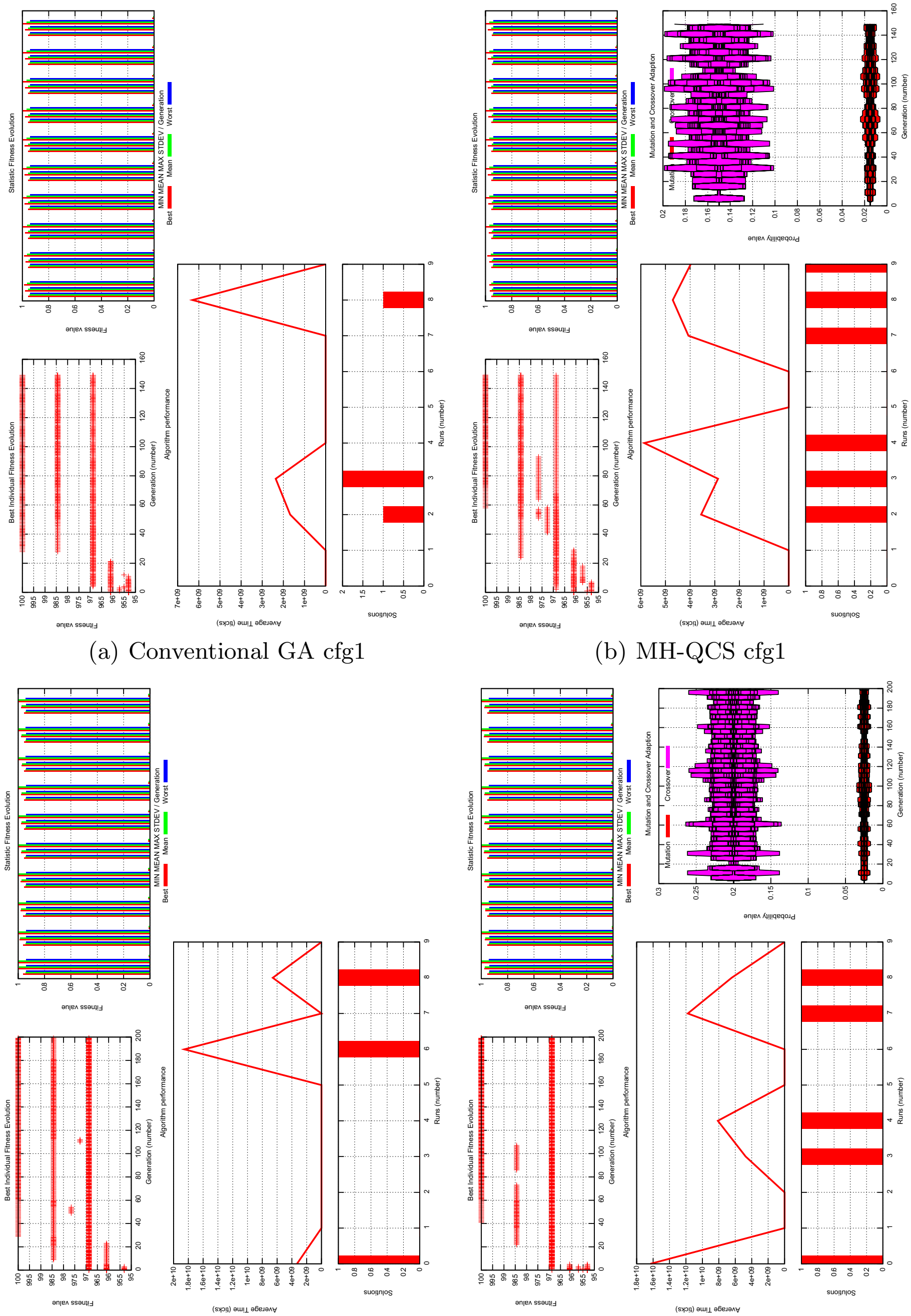
5 Experiments

To eliminate the possibility of randomness messing of the experimental results, the number of runs was set to 10, each run being independent. In the provided graphics, the average results is used for comparison purposes. The experiments were conducted on a computer with the following configuration: Intel Pentium M processor at 1.862GHz, 1GB RAM memory and Open SuSe 10.3 as operating system. During the experiments, several variables (see Table 1) were used to measure, control and manipulate the application results (looking for possible relationship situations between the sets of variables), and to create correlations between the manipulated variables and those affected by manipulation.

Table 1. Configurations Used in the Performed Experiments

Variable Name	Configuration 1	Configuration 2
Population/Generations size	100/150	150/200
Mutation type	Multiple	Single
Crossover type	Two points	One point
Selector type	Roulette Wheel	Rank
Elitism percent	0.1	0.05
Mutation/Crossover probability	0.03/0.3	0.05/0.4
Dynamic adjustment	only MH-QCS	only MH-QCS

We decided to use the "Five-Qubit EXOR"[11] as benchmark quantum circuit and to apply 4 instances for the genetic algorithm: two instances with meta-heuristic parameter control, and two normal GA instances. Using the proposed configurations, we have evolved solutions for the employed benchmark circuit. In these graphics the results are presented by using plots: statistical values are in the right-upper corner and the parameter adjustments for the genetic operators is highlighted in the right-down corner (see Figure 3).



(a) Conventional GA cfg1

(b) MH-QCS cfg1

(c) Conventional GA cfg2

(d) MH-QCS cfg2

Fig. 3. Experimental Statistic Results

6 Statistical Significance

The quality of algorithm solutions is known as the best-of-run, and in most cases it is necessary to evaluate all the intermediate results in order to render the most appropriate one [9] [13]. Our goal is to demonstrate that, in many cases, our technique (MH-QCS) performs better than a conventional GA approach, with regard to best-of-run quality. The main goal is to determine, by applying statistic significance, the highest expected best of run. Running tests for billion times each is not possible, due to a large computational effort. Thus, we need to prove our hypothesis using a small number of independent tests (in our case it is defined as 10 runs) and comparing lots of parameter settings in order to find which one performs better. Starting from the "null hypothesis" which claims that there's no difference between our algorithms, we have to compute the probability for which the null hypothesis is wrong (and we need to achieve at least 95%).

Because the number of involved population individuals is different in the used configurations, the Welch-Satterthwaite equations (see Equations 4 and 5) are used to argue that the difference between the applied algorithms is statistically significant and, at the same time, the difference value is important.

$$t_{test} = \frac{\overline{X_1} - \overline{X_2}}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \tag{4}$$

and

$$degree\ of\ freedom = \frac{\left(s_1^2/n_1 + s_2^2/n_2\right)^2}{(s_1^2/n_1)^2/(n_1-1) + (s_2^2/n_2)^2/(n_2-1)} \tag{5}$$

where $\overline{X}$ is the sample mean, n is the sample size and s^2 is unbiased estimator of the variance ($s^2 = \frac{1}{n}\sum(x_i - \overline{X})$).

By applying statistical significance over the analytical values (same values are used to generate the plots from Figure 3), our MH-QCS technique is reported to be successful in evolving quantum circuits solutions (see Table 2). The dynamic parameters customization will properly handle the objective function, the encoding and the constraints. This approach leads to a flexible genetic algorithm, where the tuning is automatically performed during the genetic evolution.

Table 2. Experimental Results

Statistical Results	MH-QCS cfg1	GA cfg1	MH-QCS cfg2	GA cfg2
Number of solutions	6	3	5	3
Average time (clocks)	4.17e+09	3.45e+09	9.48e+09	9.47e+09

7 Conclusion

This paper has presented the performance analysis for the genetic operators used by MH-QCS (the source code is available [10]), by applying statistical significance methods. Overall, the results unconditionally proved the applicability and superior effectiveness of the genetic optimization algorithms to the quantum circuit synthesis problem. The computational power overhead, required by adapting the mutation and crossover operators is reasonable small (as expressed within Table 2); moreover, the experimental results have proven significant performance improvements in comparison with a conventional GA approach in terms of convergence and number of solutions evolved.

References

1. Spector, L.: Automatic Quantum Computer Programming. In: A Genetic Programming Approach, 2nd printing edn. Springer, Heidelberg (2006)
2. Nielsen, M., Chuang, I.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2000)
3. Yao, X.: An Empirical Study of Genetic Operators in Genetic Algorithms. Microprocessing and Microprogramming 38(1-5), 707–714 (1993)
4. Hilding, F.G., Ward, K.: Automated Operator Selection on Genetic Algorithms. In: Khosla, R., Howlett, R.J., Jain, L.C. (eds.) KES 2005. LNCS (LNAI), vol. 3684, pp. 903–909. Springer, Heidelberg (2005)
5. Affenzeller, M., Wagner, S.: Offspring Selection: A New Self-Adaptive Selection Scheme for Genetic Algorithms. In: Adaptive and Natural Computing Algorithms, pp. 218–221 (2005)
6. Smit, S.K., Eiben, A.E.: Comparing Parameter Tuning Methods for Evolutionary Algorithms. In: IEEE Congress on Evolutionary Computation, pp. 399–406 (2009)
7. Wolpert, D.H., Macready, W.G.: No Free Lunch Theorems for Optimization. IEEE Transactions on Evolutionary Computation 1, 67–82 (1997)
8. Ruican, C., Udrescu, M., Prodan, L., Vladutiu, M.: A Genetic Algorithm Framework Applied to Quantum Circuit Synthesis. In: Nature Inspired Cooperative Strategies for Optimization, pp. 419–429 (2007)
9. Ruican, C., Udrescu, M., Prodan, L., Vladutiu, M.: Quantum Circuit Synthesis with Adaptive Parametres Control. In: Vanneschi, L., Gustafson, S., Moraglio, A., De Falco, I., Ebner, M. (eds.) EuroGP 2009. LNCS, vol. 5481, pp. 339–350. Springer, Heidelberg (2009)
10. Ruican, C.: Projects Web Site Page (2009), http://www.cs.utt.ro/~crys/index_files/public/qsyn.tar.gz
11. Maslov, D.: Reversible Logic Synthesis Benchmarks Page (2008), http://www.cs.uvic.ca/%7Edmaslov/
12. Gheorghies, O., Luchian, H., Gheorghies, A.: Walking the Royal Road with Integrated-Adaptive Genetic Algorithms. University Alexandru Ioan Cuza of Iasi (2005), http://thor.info.uaic.ro/~tr/tr05-04.pdf
13. Luke, S.: Essentials of Metaheuristics. Zeroth Edition (2009), http://cs.gmu.edu/~sean/book/metaheuristics/

Steering of Balance between Exploration and Exploitation Properties of Evolutionary Algorithms - Mix Selection

Adam Słowik

Department of Electronics and Computer Science, Koszalin University of Technology,
Śniadeckich 2 Street, 75-453 Koszalin, Poland
aslowik@ie.tu.koszalin.pl

Abstract. In this paper the novel selection method which can be used in any evolutionary algorithm is presented. Proposed method is based on steering between exploration and exploitation properties of evolutionary algorithms. In presented approach, at the start of the algorithm operation the probability of selection of individuals for new population is equal for all individuals. In such a case the algorithm possesses maximal value of pressure on global search of a solution space (exploration of solution space). As number of generations increases, the algorithm searches the solution space in more locally manner (exploitation of solution space) at expense of global search property. The results obtained using proposed method are compared with the results obtained using other selection methods like: roulette selection, elitist selection, fan selection, tournament selection, deterministic selection, and truncation selection. The comparison is performed using test functions chosen from literature. The results obtained using proposed selection method are better in many cases than results obtained using other selection techniques.

1 Introduction

One of the most important elements of genetic or evolutionary algorithm operation is a selection phase. Due to selection method, the new populations of potential solutions (individuals) are created. In literature, we can find many different kinds of selection techniques. Among them we can mention: roulette selection [1, 2, 3], elitist selection [4], deterministic selection [5], tournament selection [6], truncation selection [7], or fan selection [8]. Many authors try to compare the existing selection methods; for example in paper [9] a comparison of elitist selection [4], rank selection [11] and tournament selection [10] is presented. This comparison is performed with regard to selection intensity and selection variance. In paper [8], a modification of roulette selection method is presented. In this modification the survive probability of the best individual (survive schemata existing in it) is increased without guarantee that the best individual will be in the next population for sure (thus, we assure a certain random factor during selection). This modification was named as fan selection.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 213–220, 2010.
© Springer-Verlag Berlin Heidelberg 2010

The results of test functions minimization using fan selection (presented in paper [8]) have been very good in relation to the results obtained using roulette selection, and elitist selection. The fan selection presented in paper [8] depends on increase of survive probability of the best individual to the next population together with decrease of survive probability for others individuals (solutions). However, such approach guarantee, that the best individual will be selected to the new population more often than others individuals. Therefore, the strong loss of solutions diversity occurs in the population. This negative effect is named as super-individual problem, and can be accepted only in optimization of single-modal functions, where the risk of stick in local extreme does not exist. Also, we have noticed, that in the fan selection (for small values of parameter a) the value of fitness function increases for worse individuals in expense of average quality individuals [12], what is a negative effect. Nowadays, guarantee of balance between exploration property (whole solution space is searched), and exploitation property (solution space near the best solution is searched) is important during design of selection operators. Therefore, it is required to search the solution space globally (at the start of algorithm), and together with increase of number of generations the factor of local search (around the best solution) is more important.

In this paper, the mix selection is presented. In proposed selection the exploration property and exploitation property are balanced using adaptive changes of α parameter (see section 2). The proposed method has been tested using test functions chosen from literature. The results obtained using mix selection are compared with results obtained using other selection methods.

2 Mix Selection

The proposed selection method depends on modification of α parameter ($\alpha \in$ [-1; 1]) values of relative fitness *rfitness* for particular individuals:

- when $\alpha \geq 0$

- for the best individual

$$rfitness'_{max} = rfitness_{max} + (1 - rfitness_{max}) \cdot \alpha \tag{1}$$

- for others individuals

$$rfitness' = rfitness \cdot \left(\frac{rfitness_{max} - rfitness'_{max}}{\sum_{i=1}^{M} rfitness_i - rfitness_{max}} + 1 \right) \tag{2}$$

- when $\alpha < 0$

  - for all individuals

$$rfitness' = rfitness + \left(rfitness - \frac{1}{M} \right) \cdot \alpha \tag{3}$$

where: $rfitness'_{max}$-new relative fitness of the best individual; $rfitness_{max}$-old relative fitness of the best individual; α-scaling factor $\alpha \in [-1; 1]$; $rfitness'$-new relative fitness of chosen individual; $rfitness$-old relative fitness of chosen individual; M-number of individuals in population.

In the mix selection, the best solution (individual) among all the best solutions found hitherto is remembered. At the moment, when the best solution existing in population is lost after selection process, then remembered solution is inserted in the place of actual the best solution in population (in order to do not increase the selection intensity). Inserting the best solution found so far into place of the best solution in current population is needed in order to prevent the loss diversity in the population. The loss of diversity in the population can be caused when the best solution will be inserted in the place of the worst individual. This operation also prevents destroying the best solution found so far.

In the proposed method of mix selection, the exploration and exploitation properties are changed with respect to value of α parameter. When $\alpha = -1$, then all individuals possesses identical chances to be selected to the new population (Fig. 1a). The selection intensity is equal to zero, and the proposed method possesses exploration property (whole potential solution space is searched - the diversity of individuals in the population is not lost). Together with increase of α parameter value (Fig. 1b) up to the value $\alpha = 0$ (Fig. 1c), the exploration properties are decreased. When value of parameter $\alpha = 0$, then all individuals possess gradually unmodified values of relative fitness (identically as in roulette selection method). Together with successive increasing of α values (Fig. 1d and

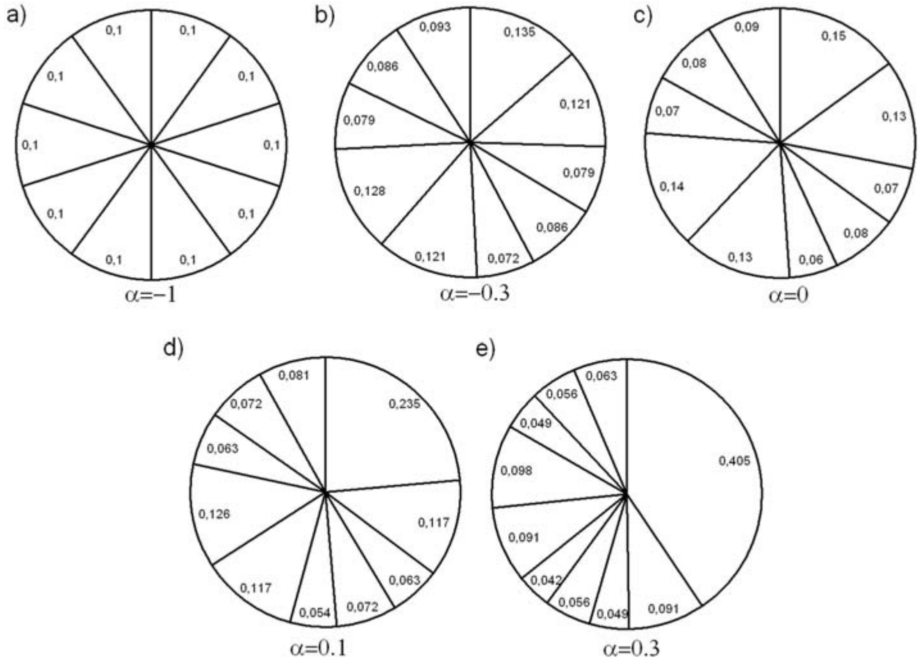


Fig. 1. Illustration of mix selection process for different values of α parameter: $\alpha = -1$ (a), $\alpha = -0.3$ (b), $\alpha = 0$ (c), $\alpha = 0.1$ (d), $\alpha = 0.3$ (e)

Fig. 1e) the exploitation properties are increased in proposed selection method (searching of solutions space close to the best solution - the solution diversity is decreased in population). For $\alpha = 1$, the best individual is only selected to the new population. Then the proposed method possesses maximal selection intensity. Therefore, it can be seen, that together with changes of α parameter values the properties of proposed selection method are changed too.

The additional advantage of proposed method is its generality. Due to mix selection, we can determine the relations between global searching and local searching (decreasing or increasing of selection intensity). In the mix selection method, an adaptive change of parameter α value is introduced. Due to this operation, we can vary the properties of proposed method and adapt mix selection to the solving problem.

3 Adaptive Change of Parameter α Value

In the case of optimization tasks in continuous domain, it is necessary to use specialized mutation operators (for example the non-uniform mutation [1, 13]) in genetic and evolutionary algorithms. These operators search a whole potential solution space at the start, and while the number of generation increases the searching is more and more local (near the best solution). This conception is also used in the adaptive changes of parameter α values in mix selection. It is assumed, that evolutionary algorithm is operating by T_{max} generations (identically as in non-uniform mutation [1, 13]), and at the beginning the solution space should be globally searched and for generations near T_{max} the solution space should be locally searched (near the best solution). In presented in this paper version of the mix selection, the linear variation of the parameter α is assumed as follows:

$$\alpha = \frac{2}{T_{max}} \cdot Iter - 1 \tag{4}$$

where: T_{max}—maximal number of iteration of the algorithm, $Iter$—current number of iteration.

The proposed mix selection possesses exploration property (more global search of the solution space) in the range from 0 to $T_{max}/2$ generations. For further generations, the exploitation property (more local search of the solution space) becomes more strong.

4 Assumed Test Functions

Following test function chosen from literature [14, 15] have been taken to quality verification of proposed method (GM represents the global minimal value, n represents number of variables in optimized function). These functions are widely used as benchmarks to test the quality of evolutionary algorithms. In all test functions minimization problems are considered.

- De Jong function F1

$\sum_{i=1}^{n} x_i^2$; $-100 \leq x_i \leq 100$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Ackley function F2

$20 - 20 \cdot \exp\left(-0.2 \cdot \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}\right) + \exp(1) - \exp\left(\frac{1}{n} \cdot \sum_{i=1}^{n} cos\left(2 \cdot \pi \cdot x_i\right)\right)$; $-100 \leq x_i \leq 100$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Griewank function F3

$\frac{1}{4000} \cdot \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ $-600 \leq x_i \leq 600$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Rastrigin function F4 $10 \cdot \mathrm{n} + \sum_{i=1}^{n} \left(x_i^2 - 10 \cdot cos\left(2 \cdot \pi \cdot x_i\right)\right)$

$-500 \leq x_i \leq 500$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 20$

- Schwefel function F5

$418.9828872724339 \cdot \mathrm{n} - \sum_{i=1}^{n} \left(x_i \cdot sin\left(\sqrt{|x_i|}\right)\right)$

$-500 \leq x_i \leq 500$; GM=0 in $(x_1, x_2, ..., x_{30}) = (420.96874636, ..., 420.96874636)$; $n = 30$

- High Conditioned Elliptic function F6

$\sum_{i=1}^{n} \left(10^6\right)^{\frac{i-1}{n-1}} \cdot x_i^2$;$-5 \leq x_i \leq 5$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Non-Continuous Rastrigin function F7

$\sum_{i=1}^{n} \left(y^2 - 10 \cdot cos\left(2 \cdot \pi \cdot y_i\right) + 10\right)$; $y_i = \begin{cases} x_i, & when \ \ |x_i| < 0.5 \\ round\left(2 \cdot x_i\right)/2, & when \ \ |x_i| \geq 0.5 \end{cases}$

$-500 \leq x_i \leq 500$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Non-Continuous Expanded Schaffer function F8

$F\left(y_1, y_2\right) + F\left(y_2, y_3\right) + ... + F\left(y_{n-1}, y_n\right) + F\left(y_n, y_1\right)$; $F\left(x, y\right) = 0.5 + \frac{\left(sin^2\left(\sqrt{x^2+y^2}\right) - 0.5\right)}{\left(1+0.001 \cdot \left(x^2+y^2\right)\right)^2}$

$y_i = \begin{cases} x_i, & when \ \ |x_i| < 0.5 \\ round\left(2 \cdot x_i\right)/2, & when \ \ |x_i| \geq 0.5 \end{cases}$

$-500 \leq x_i \leq 500$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Rotated Expanded Schaffer function F9

$F\left(x_1, x_2\right) + F\left(x_2, x_3\right) + ... + F\left(x_{n-1}, x_n\right) + F\left(x_n, x_1\right)$; $F\left(x, y\right) = 0.5 + \frac{\left(sin^2\left(\sqrt{x^2+y^2}\right) - 0.5\right)}{\left(1+0.001 \cdot \left(x^2+y^2\right)\right)^2}$

$-500 \leq x_i \leq 500$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- De Jong function F10

$\sum_{i=1}^{n} i \cdot x_i^4$; $-100 \leq x_i \leq 100$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Bohachevsky function F11

$\sum_{i=1}^{n} \left(x_i^2 + 2 \cdot x_{i+1}^2 - 0.3 \cdot cos\left(3 \cdot \pi \cdot x_i\right) - 0.4 \cdot cos\left(4 \cdot \pi \cdot x_{i+1}\right) + 0.7\right)$

$-15 \leq x_i \leq 15$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Rosenbrock function F12

$\sum_{i=1}^{n} \left(100 \cdot \left(x_i^2 - x_{i+1}\right)^2 + \left(x_i - 1\right)^2\right)$

$-5 \leq x_i \leq 5$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Scaled Rastrigin function F13

$10 \cdot n + \sum_{i=1}^{n} \left(\left(10^{\frac{i-1}{n-1}} \cdot x_i\right)^2 - 10 \cdot cos\left(2 \cdot \pi \cdot 10^{\frac{i-1}{n-1}} \cdot x_i\right)\right)$

$-5 \leq x_i \leq 5$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Skew Rastrigin function F14

$10 \cdot n + \sum_{i=1}^{n} \left(y_i^2 - 10 \cdot cos\left(2 \cdot \pi \cdot y_i\right)\right)$; $y_i = \begin{cases} 10 \cdot x_i, \ when \ x_i > 0 \\ x_i, \quad\ \ otherwise \end{cases}$

$-5 \leq x_i \leq 5$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

- Schaffer function F15

$\sum_{i=1}^{n-1} \left(x_i^2 + x_{i+1}^2\right)^{0.25} \cdot \left[sin^2\left(50 \cdot \left(x_i^2 + x_{i+1}^2\right)^{0.1}\right) + 1\right]$

$-100 \leq x_i \leq 100$; GM=0 in $(x_1, x_2, ..., x_{30}) = (0, 0, ..., 0)$; $n = 30$

5 Description of Experiments

The experiments were performed using test functions presented in section four. In evolutionary algorithm, the following parameters were assumed: individuals were coded as a real-number strings (each gene represents one variable in optimized function), T_{max}=1000, M=50, probability of crossover = 0.7, probability of mutation = $\frac{1}{n}$. The individuals in population were created randomly. Simple one-point crossover [1] was taken as a crossover operator. The crossover operator depend on cutting of two randomly chosen individuals from population in one randomly chosen point, and then the cut fragments of chromosomes are exchanged between them. The non-uniform mutation [1, 13] was taken as a mutation operator. This operator is responsible for precise tuning up of the evolutionary algorithm to potential solution; the value of mutated gene is computed as follows:

$$x'_k = \begin{cases} x_k + \mathit{\Delta}\left(t, x_{k,max} - x_k\right), \; when \; q < 0.5 \\ x_k - \mathit{\Delta}\left(t, x_k - x_{k,min}\right), \; when \; q \geq 0.5 \end{cases} \tag{5}$$

$$\mathit{\Delta}\left(t, y\right) = y \cdot r \cdot \left(1 - \frac{t}{T_{max}}\right)^b \tag{6}$$

where: x_k−value of k−th gene randomly chosen for mutation $x_k \in [x_{k,min}, x_{k,max}]$, x'_k−value of k−th gene after mutation, q and r− random values from the range [0, 1], t−current number of generation, T_{max}−maximal number of generation, b−level of inhomogeneity (typically b=2). Function $\mathit{\Delta}\left(t, y\right)$ takes values from the range [0, y], and the probability, that $\mathit{\Delta}\left(t, y\right) \approx 0$ increases together with increase of t value. This property causes, that this mutation operator is searching the whole solution space uniformly at the start of its operation. In next stages (when t is higher and higher) only local surrounding is searched using this operator.

During the operation of the evolutionary algorithm only selection operator is changed. The parameter a=0.3 is taken for fan selection (identically as in paper [8]). The size of tournament group equal to 2 is assumed for tournament selection. Truncation threshold equal to 0.5 is assumed for truncation selection. In experiments, the evolutionary algorithm was executed 25 times for each test function. In Table 1, average values and standard deviations for the best results obtained after 25-fold repetition of evolutionary algorithm for particular selection methods are presented. The symbols in Table 1 are as follows: SM-chosen selection method, RO-roulette selection, EL-elitist selection, FAN-fan selection, TOU-tournament selection, DET-deterministic selection, TRU-truncation selection, MIX-mix selection.

It can be seen from Table 1, that in many cases results obtained using mix selection are better than results obtained using other selection methods.

In order to check an importance of results presented in Table 1 t-Student statistical test (with 48 degrees of freedom) has been performed for all combinations between results obtained using mix selection and results obtained using other selection methods. The obtained results are not presented in this paper

Table 1. Average values of the best results obtained after 25-fold repetition of evolutionary algorithm for each selection method

SM	F1	F2	F3	F4	F5
RO	327.90±98.11	4.75±0.37	4.44±0.84	3600±1336	1161.83±361.00
EL	166.54±40.39	3.33±0.32	2.53±0.34	1524±430	259.13±96.72
FAN	0.96±0.32	0.31±0.05	0.79±0.14	77.19±18.08	2.64±1.32
TOU	94.64±23.57	3.36±0.31	1.84±0.15	878.53±213.78	383.29±170.21
DET	318.05±108.90	3.81±0.35	3.82±1.05	4379±3126	562.16±150.69
TRU	47.96±11.91	2.79±0.24	1.48±0.14	607.69±161.83	212.91±136.55
MIX	0.37±0.17	0.19±0.04	0.50±0.14	43.16±9.37	95.87±94.91
SM	F6	F7	F8	F9	F10
RO	53715±19365	9938±2572	9.90±0.75	10.09±0.62	2280256±1106167
EL	6732±2516	4344±858	7.13±0.86	6.71±0.78	460723±268790
FAN	45.15±28.52	116.00±28.53	3.18±0.61	3.34±0.64	3.37±2.00
TOU	605±163	2760±741	7.57±0.63	7.57±0.52	33973.05±20173.35
DET	549057±4455	7670±3215	7.31±0.86	7.71±0.77	32067286±51090369
TRU	295±121	1443±321	6.78±0.98	6.57±0.70	10216.88±4766.51
MIX	16.12±9.77	65.22±14.73	2.81±0.43	2.93±0.69	0.29±0.18
SM	F11	F12	F13	F14	F15
RO	40.74±7.35	347.57±91.61	137.54±18.96	129.10±18.71	65.33±8.65
EL	25.99±3.44	75.92±31.57	96.77±11.21	87.43±11.74	55.14±5.82
FAN	0.95±0.33	75.92±31.57	8.45±2.11	4.64±1.60	24.35±3.33
TOU	19.82±2.32	137.39±33.87	67.53±9.17	50.30±5.97	49.02±4.84
DET	38.41±7.72	615±568	180.23±97.36	233.48±98.43	49.49±4.21
TRU	14.27±2.37	147.16±42.15	53.82±8.21	42.67±8.09	44.36±4.26
MIX	0.49±0.29	58.96±36.64	5.12±1.89	3.57±1.54	18.37±2.46

with respect to space limitation. But in all cases (besides one case), the results obtained using proposed method are statistically important (with 95% degree of trust).

6 Conclusions

In this paper, the novel selection method named mix selection is presented. In this method at the start of the algorithm operation the whole solution space is searched globally, and together with increase of the number of generations the solution space is searched more and more locally. Proposed method has been tested using test functions chosen from literature. Results obtained using mix selection have been compared with results obtained using other selection methods. In many cases, the results obtained using proposed approach are better or comparable with the results obtained using other methods. Also, the results obtained using mix selection are statistically important (their importance has been confirmed using statistical t-Student test). The main advantage of the proposed method is a possibility of defining relations between global and local searches of potential solution space. Also, the disadvantage existing in fan selection (which has been described in introduction) has been eliminated in proposed method. The mix selection can be used in combinatorial problems too.

References

1. Michalewicz, Z.: Genetic algorithms + data structures = evolution programs. Springer, Heidelberg (1992)
2. Goldberg, D.E.: Genetic algorithms in search, optimization and machine learning. Addison-Wesley Publishing Company Inc., New York (1989)
3. Arabas, J.: Lectures on evolutionary algorithms, WNT (2001) (in Polish)
4. Zen, S., Zhou Yang, C.T.: Comparison of steady state and elitist selection genetic algorithms. In: Proceedings of 2004 International Conference on Intelligent Mechatronics and Automation, August 26-31, pp. 495–499 (2004)
5. Takaaki, N., Takahiko, K., Keiichiro, Y.: Deterministic Genetic Algorithm. In: Papers of Technical Meeting on Industrial Instrumentation and Control, IEE Japan, pp. 33–36 (2003)
6. Blickle, T., Thiele, L.: A Comparison of Selection Schemes used in Genetic Algorithms. Computer Engineering and Communication Networks Lab, Swiss Federal Institute of Technology, TIK Report, No. 11, Edition 2 (December 1995)
7. Muhlenbein, H., Schlierkamp-voosen, D.: Predictive Models for the Breeder Genetic Algorithm. Evolutionary Computation 1(1), 2549 (1993)
8. Slowik, A., Bialko, M.: Modified Version of Roulette Selection for Evolution Algorithm - The Fan Selection. In: Rutkowski, L., Siekmann, J.H., Tadeusiewicz, R., Zadeh, L.A. (eds.) ICAISC 2004. LNCS (LNAI), vol. 3070, pp. 474–479. Springer, Heidelberg (2004)
9. Blickle, T., Thiele, L.: A Comparison of Selection Schemes used in Evolutionary Algorithms. Evolutionary Computation 4(4), 361–394 (1996)
10. Miller, B.L., Goldberg, D.E.: Genetic algorithms, tournament selection and the effects of noise. Complex Systems 9, 193–212 (1995)
11. Bäck, T., Hoffmeister, F.: Extended Selection Mechanisms in Genetic Algorithms. In: Belew, R.K., Booker, L.B. (eds.) Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 92–99. Morgan Kaufmann Publishers, San Mateo (1991)
12. Private communication from Prof. Wojciech Jedruch, Department of Electronics, Telecommunications and Informatics. Gdansk University of Technology (2007)
13. Zhaoa, X., Gaob, X.-S., Hu, Z.-C.: Evolutionary programming based on non-uniform mutation. Applied Mathematics and Computation 192(1), 1–11 (2007)
14. Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.P., Auger, A., Tiwari, S.: Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization. Technical Report, Nanyang Technological University, Singapore And KanGAL Report Number 2005005 (Kanpur Genetic Algorithms Laboratory, IIT Kanpur) (May 2005)
15. Hansen, N., Kern, S.: Evaluating the CMA Evolution Strategy on Multimodal Test Functions. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 282–291. Springer, Heidelberg (2004)

Extending Genetic Programming to Evolve Perceptron-Like Learning Programs

Marcin Suchorzewski

West Pomeranian University of Technology
Faculty of Computer Science and Information Technology
ul. Żołnierska 49, 71-210 Szczecin, Poland
msuchorzewski@wi.zut.edu.pl

Abstract. We extend genetic programming (GP) with a local memory and vectorization to evolve simple, perceptron-like programs capable of learning by error correction. The local memory allows for a scalar value or vector to be stored and manipulated within a local scope of GP tree. Vectorization consists in grouping input variables and processing them as vectors. We demonstrate these extensions, along with an island model, allow to evolve general perceptron-like programs, i.e. working for any number of inputs. This is unlike in standard GP, where inputs are represented explicitly as scalars, so that scaling up the problem would require to evolve a new solution. Moreover, we find vectorization allows to represent programs more compactly and facilitates the evolutionary search.

Keywords: Genetic programming; evolutionary neural networks; learning programs; supervised learning.

1 Introduction

Genetic Programming (GP) is an evolutionary algorithm which evolves computer programs. Since it's formulation in early '90, GP has delivered a number of human-competitive results in various problem domains [4]. Yet, among a wide range of innovative techniques and applications proposed for GP, we don't find many specifically concerned with evolving learning programs. A comprehensive review of state of art in GP can be found in [7].

From a biological perspective, learning and evolution are both adaptations, though operating on different time scales. Evolution adapts individuals to slowly changing environmental conditions, along many generations. Learning in turn, allows to adapt individuals to conditions changing during their lifetime. A common characteristic of biological and artificial adaptations is that stationary conditions of the environment/problem are reflected in structure, while more dynamic changes are reflected in changing parameters. The vast majority of research in GP concentrates on evolving structures only, which are capable of solving static or dynamic problems, but not capable to learn, i.e. to solve either new instances of the problem, or to respond quickly to changing conditions while

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 221–228, 2010.
© Springer-Verlag Berlin Heidelberg 2010

solving their task. Perhaps the major obstacle to evolve learning programs using GP lies in that learning parameters requires memory and memory constructs in GP have a bad reputation of being tough to evolve. A recent critique of GP [11] exposed some weaknesses of GP in evolving structures more complex than arithmetic expressions, such as loops and modules. This is also relevant to memory. Perhaps that's why so much more research has been devoted to evolving neural networks, for which many learning algorithms are readily available, than to evolve programs, which have to discover their own means to learn.

The concept of learning programs is in fact very closely related to evolutionary neural networks, which has already attracted much attention. We find many approaches here: evolving topologies, which further require external learning, evolving weighted topologies which do not learn, evolving learning rules for predefined topologies or using evolutionary algorithm just to find weights or fine-tune other parameters of neural networks. Seldom attempted, however, is to evolve structure and learning algorithm simultaneously. See [12,5,2] for an overview of the topic.

In this paper we propose two new extensions to GP which allow to evolve "autonomously learning" programs, i.e. without any help of external learning algorithm. Learning is embedded in the program and requires no "extra mode" of execution. For now, we demonstrate the capabilities of the extended GP on a supervised learning classification task, with two linearly separable classes. The only requirement for the evolved solution to work properly is to include classification error of the previous sample in the inputs to the program. Evolved solution programs discovered a simple rule to ignore the error when it's zero, and adapt parameters when the error is non-zero. GP extended with local memory can — and often does — evolve structures which are functionally equivalent to perceptron neurons learned via delta rule [10].

2 Extending GP

2.1 Local Memory

Memory is a necessary ingredient of learning. There are three well known types of memory in GP: named memory, automatically defined storage (ADS) [3] and indexed memory [8], and each of them have some shortcomings. Named memory requires inclusion of separate node types for each new memory cell to be used by a program. The two other constructs also require some predefined parameters. ADS and indexed memory function nodes require additional indexing arguments, which increase the representational intricacy. And finally, all these constructs are inherently global, i.e. the memory can be read and write from anywhere in the program. The local memory proposed in this paper is a simple construct, which neither requires indexing nor needs to be anyhow predefined. Although we didn't perform any empirical comparison yet, we suppose the property of locality might also be beneficial for evolvability of the programs.

To implement the local memory in GP trees we use two special nodes: 1-argument function `SetM` and the terminal `GetM`. Node `SetM` contains data, while

`GetM` holds only a pointer to data. These nodes can be freely moved across the program tree by mutation or crossover operators during reproduction phase, but in order to work properly, the memory must be initialized before the first evaluation. This is done in one traversal of the tree in prefix order (i.e. the root first). Each `GetM` node is connected to its first `SetM` ancestor on the path to the root, though only if there is any. All values held in `SetM` nodes are initialized to 0. After the memory has been connected and initialized, it works as follows. The argument passed to the `SetM` node is returned unchanged, though it is stored. `GetM` terminal node returns the value stored in a `SetM` to which it is connected, or 0 if it's not connected.

We called this construct a "local memory" because the `SetM` and `GetM` nodes form a memory which is operational only in some fragment of the tree, i.e. it has limited scope. Moving a subtree containing `SetM` node to another position or another tree, automatically results in moving all the relevant `GetM` nodes also, although the inverse is not necessarily true. Moreover, local memory doesn't need explicit addressing or indexing, as is the case with named memory, indexed memory or ADSs.

2.2 Vector Processing

While memory is almost necessary for a program to learn, what makes the evolved solutions general with respect to the dimensionality is vectorization, i.e. conversion of a program from scalar into vector implementation, where single instruction operates on multiple data. Vectorization is not always possible to accomplish, however in our scenario we assume relatively specific problem and the function set for it, and therefore it's quite straightforward. First, we assume we are solving a problem characterized by two groups of inputs: a single scalar input and varying number of "ordinary" input variables. These ordinary input variables are grouped into a vector of length N, which is a dimensionality of the problem.

To obtain a system closed in terms of type consistency and evaluation safety [7], it's necessary for all functions used in the program to be well defined for any combination of vector and scalar arguments. This condition is obviously satisfied for addition and multiplication operations, assuming there is only a single distinct vector dimensionality in the program. Also the local memory nodes can be well defined for scalars and vectors, whatever happens to be the argument of the `SetM` node.

A minor issue follows from the fact, that having a vector somewhere in the leaves of the tree, we always have a vector on the output. An obvious workaround would be to sum up the vector and to return scalar value as a proper classification result. This would be a kind of wrapper put on the program. Much more autonomous solution would be to allow for the program to convert vectors into scalars itself, by means of summation (σ node). The program returning vector as the output can be either assigned poor fitness, or the vector might be summed up in a kind of wrapper. We prefer the first option by default.

Vector processing is certainly not a completely new concept in GP. It's closely related to the *strongly typed GP* [6]. We are not aware, however, of any previous research demonstrating its importance in achieving generality of the solution. Perhaps the generality obtained with vectorization might also be obtained using developmental GP. Yet still, we find vectorization is simpler and yields more compact solutions than those possibly generated with developmental encoding.

2.3 A Note on Island Model

An important feature of vectorized programs is that they are more general than scalar ones. This means, that program solution for N_1-dimensional problem is likely to be a solution for N_2-dimensional, also. To verify this claim we use a variant of an island model of population structure [9]. The special feature of our variant is that each population in the "array of islands" solves a problem of different dimensionality. Populations evolve in parallel and in each generation some fraction of individuals is exchanged between two randomly selected islands. The fraction is determined by the migration rate parameter. Obviously, when the migration rate is 0, the island model is equivalent to standard GP with a single population, except it's run in parallel.

3 Experiment

3.1 Problem Setup

In order to demonstrate that extended GP system allows to evolve learning programs, we use the following artificial test problem. Let $\mathbf{x} \in \mathbb{R}^N$ be a point (vector) drawn from a multivariate normal distribution with mean 0 and standard deviation 1 in all dimensions. Let $X = \{\mathbf{x}_i; i = 1, \ldots, 20\}$ denote training set of 20 points. Now, let draw another N-dimensional vector $\mathbf{w}$ from the same multivariate normal distribution, which will act as a hyperplane bisecting the space into 2 classes. For each vector $\mathbf{x}_i$ calculate its class using scalar product:

$$y_i = (\mathbf{w} \cdot \mathbf{x}_i > 0) \ , \tag{1}$$

assigning 0 to points lying on the one side of hyperplane and 1 to those on the other. Next, we draw a translation vector $\mathbf{z}$ from the same distribution and translate all points from X by that vector. As a result we obtain a set of 20 randomly labeled, yet linearly separable points, distributed normally around some point near the origin of a coordinate system.

To evaluate a candidate program, we proceed as follows:

```
For 8 epochs do
  Permute points in $X$, and class labels in $Y$ correspondingly
  Set error $e := 0$
  For each point $\mathbf{x}_i$ in $X$ do
    Calculate the output of the program $a := P(e, \mathbf{x}_i)$
    Calculate new error $e := y_i - (a > 0.5)$, which is either -1, 0, or 1
```

The above algorithm makes a trial. From each trial we calculate the average number of misclassifications, but taken from the last epoch only (not shown). If the program classifies all inputs correctly in some epoch, the trial is interrupted with perfect score 0. To fully evaluate the candidate program, we perform 10 trials, using new training sets each time. In each generation, the best individual in the population is tested not for 10 trials, but 200 trials, whether it qualifies as a solution. Note we do not perform any generalization tests or validation, because we are only interested in basic learning capability, so it's sufficient to observe learning on the training set only.

Finally, we shall define what a solution is. Since the programs have limited number of epochs to learn, some trials are likely to end with imperfect score, and this is also relevant to conventional perceptron learning. Since we would be satisfied having the performance of perceptron, we measured mean errors of independent implementation of perceptron learning on our task and took them as a standard fitness thresholds for a program to count as a solution. These thresholds for $N = [1\ 2\ 4\ 7\ 11\ 16]$ amount to $[0.067\ 0.066\ 0.068\ 0.060\ 0.056\ 0.040]$ respectively. It means for example, that in the 1-dimensional case, about 6.7% of samples were still misclassified in the 8th epoch. Apparently, in terms of rate of convergence the problem gets easier as N approaches the size of training set.

3.2 GP Setup

Although our GP algorithm implementation doesn't strictly follow any canonical algorithm, we believe it's not necessary in our research. This is because we evaluate our approach in qualitative terms rather than quantitative. In particular, we compare the increase of time complexity required to solve the problem as it scales-up in the number of inputs. In Table 1 we list some basic GP parameters.

Table 1. GP parameters tableau

Objective	Learn to classify 20 data points from N-dimensional space
Population	1024
Functions	+, -, *, $\mathtt{z}^{-1}$ (1-step delay), SetM, GetM
Terminals	e, ephemeral normal rand. const. $\mathtt{c} \sim N(0,1)$, inputs $\mathtt{u}_1 \ldots \mathtt{u}_N$
Std fitness J	Misclassification frequency per data point (minimization)
Eff fitness F	Using parsimony: $F = J + 0.001 * tree\_size$
Selection	Tournament size 5, generational, non-elitist
Initialization	Grow with probability 0.6, up to 5 levels
Mutation	$P_m = 0.4$, random subtree, grown with prob. 0.6 up to 3 levels
Crossover	$P_c = 0.6$, one-child, independent from mutation
Termination	200 generations or solution found
Migration rate	0% or 1%

In this paper we perform 9 experiments, counting 20 evolutionary runs each (series). Each experiment corresponds to a bit different GP setup, though the basic algorithm is fixed. Each run is performed using an array of 6 populations, each solving the classification problem having different dimensionality $N = 1, 2, 4, 7, 11, 16$. We examine the following setups, which we explain below:

- Std — basic GP algorithm with no migration between populations,
- Std-m — basic GP with migration,
- H, H-m — GP using H function instead of $\mathtt{z}^{\mathtt{-1}}$, SetM and GetM nodes,
- ADF, ADF-m — GP using automatically defined functions (ADFs),
- X, X-m — using vectorization, i.e. v terminal representing $\mathbf{x}_i$ vectors, instead of $\mathtt{u_1} \ldots \mathtt{u_N}$ input nodes, using also S function to sum up vectors,
- X-m-S — like X-m, except if the result of the program is vector, it's automatically summed up (wrapped).

The H function node has two arguments and also internal storage for weight w and for previous argument x, and is evaluated as follows:

$$w_t := w_{t-1} + 0.1 x_{t-1} y \tag{2}$$

$$H(x, y) := w_t x \ . \tag{3}$$

As we can see, the H function node implements Hebb's rule of learning, which in turn becomes delta rule, if we assume y is the error. Thus H function constitutes almost a perfect module to be exploited by the algorithm. The reason we examine this variant of GP is to see how the algorithm will behave when it has perfect module at its disposal, while still using all the $\mathtt{u_i}$ input nodes in scalar form.

The ADF and ADF-m variants of GP employ standard technique of creating ADFs, described in [3]. The predefined number of ADFs is 1, with randomly chosen number of arguments, and possible *deletion by expansion*. The aim of these two experiments is to inquire how easy it is for GP with ADFs to discover useful modules for the problem.

3.3 Results

The results of experiments are presented in Table 2, in terms of average number of generations that must be evaluated to obtain a solution. The measure is very similar to Koza's computational effort, except simpler to calculate. Additionally, we find it more stable, when the number of simulation runs is low (see [1]). For a problem instance n:

$$E_n = \text{\#generations\_evaluated\_in\_all\_runs} \ / \ \text{\#solutions\_found} \ . \tag{4}$$

As we can see, both standard GP setups found solutions only for the simplest case of 1-dimensional problem. Having perfect module at hand, however, it can find solutions much more easily, even for highest dimensionality. Note, however, that time complexity grows faster than linearly here. Although GP can make efficient use of the perfect module, it has a big trouble in finding it, which is evident in ADF experiments. Finally, we note two interesting results regarding vectorized GP with migration. First, finding the solution for all $N > 1$ problem instances is almost equally difficult, and second, programs can take advantage

Table 2. Average number of generations to yield a solution

	Problem dimensionality N					
GP setup	1	2	4	7	11	16
Std	139	–	–	–	–	–
Std-m	328	–	–	–	–	–
H	5	12	29	55	104	294
H-m	4	13	29	61	112	319
ADF	126	–	–	–	–	–
ADF-m	162	–	–	–	–	–
X	140	1273	–	–	–	–
X-m	90	739	943	750	623	621
X-m-S	93	119	121	113	103	99

from communication between populations solving different problem instances. In fact, the usual pattern we observed when migration was enabled, is that good programs were usually found for 1-dimensional instance first, and then due to migration they biased the search in other populations, resulting in quick improvements and finally the solution. Sometimes however, the order of succession was opposite, and perhaps that's why the effort is lower for 1-dimensional instance when the migration is enabled. The last remark concerns the summation function node. Evidently, the effort is several times higher, when programs have to sum up vectors by themselves, although the time complexity grows equally for X-m and X-m-S setups.

In Figure 1 we present an example of an evolved program. It comes from a population solving 4-dimensional problem, yet it's also a solution for all the other instances. The program is probably the minimal general solution, though even smaller solutions were observed for 11- and 16-dimensional instances. The root of the tree is on the left. Branch denoted with (1) holds the parameter (weight) vector and calculates its update. Branch (2) acts as a kind of bias, and branch (3) is where inputs are actually weighted to perform classification. The program has an obvious analogy to perceptron neuron, however it's not functionally identical.

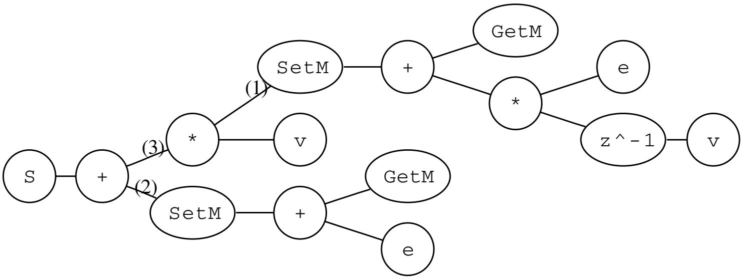


Fig. 1. Evolved program being a general solution for the classification problem

4 Summary and Conclusions

In the paper we proposed a new kind of memory for GP, that allows programs to store and manipulate parameters. We demonstrated how vectorization can be utilized to evolve compactly represented solutions capable of learning to perform binary classification task. Evolved solutions are general with respect to dimensionality of the problem. Vectorized representation allowed the evolution to take advantage of island model and communication between populations solving several instances of the classification problem. We showed that the computational effort required to solve the problem is constant with increasing problem dimensionality, which is obviously in contrast to non-vectorized representation.

Benefits provided by vectorization were demonstrated for a problem in which a single group of inputs was obviously susceptible to vector processing, i.e. the inputs were indiscernible and each input required the same sequence of computation for a program to work properly. Vector processing could not be so straightforward if the problem involved many groups of inputs and each group had possibly different cardinality. It remains for future work to investigate how the extensions could work in such cases. Beside that, we plan to generalize our approach to a wider range of learning problems, such as non-linearly separable classification or unsupervised learning and test it on real-world problems.

References

1. Christensen, S., Oppacher, F.: An Analysis of Koza's Computational Effort Statistic for Genetic Programming. In: Foster, J.A., Lutton, E., Miller, J., Ryan, C., Tettamanzi, A.G.B. (eds.) EuroGP 2002. LNCS, vol. 2278, pp. 182–191. Springer, Heidelberg (2002)
2. Floreano, D., Dürr, P., Mattiussi, C.: Neuroevolution: From Architectures to Learning. Evolutionary Intelligence 1(1), 47–62 (2008)
3. Koza, J.R.: Genetic Programming III: Darwinian Invention and Problem Solving. Morgan Kaufmann Pub., San Francisco (1999)
4. Koza, J.R., Keane, M.A., Streeter, M.J.: What's AI Done for Me Lately? Genetic Programming's Human-Competitive Results. IEEE Intell. Syst., 25–31 (2003)
5. Miikkulainen, R.: Evolving Neural Networks. In: Proc. of the 2007 GECCO Conf. Comp. on Genetic and Evol. Comput., pp. 3415–3434. ACM, New York (2007)
6. Montana, D.J.: Strongly Typed Genetic Programming. Evolutionary Computation 3(2), 199–230 (1995)
7. Poli, R., Langdon, W.B., McPhee, N.F.: A Field Guide to Genetic Programming. Lulu Press (2008)
8. Teller, A.: Turing Completeness in the Language of Genetic Programming with Indexed Memory. In: Proc. of the 1994 IEEE World Congr. on Comput. Intell., vol. 1, pp. 136–141 (1994)
9. Tomassini, M.: Spatially Structured Evolutionary Algorithms. Springer, Heidelberg (2005)
10. Widrow, B., Lehr, M.A.: 30 Years of Adaptive Neural Networks: Perceptron, Madaline and Backpropagation. Proc. IEEE 78(9), 1415–1442 (1990)
11. Woodward, J.R., Bai, R.: Why Evolution Is Not a Good Paradigm for Program Induction: A Critique of Genetic Programming. In: Proc. of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, pp. 593–600 (2009)
12. Yao, X.: Evolving Artificial Neural Networks. Proc. IEEE 87(9), 1423–1447 (1999)

An Informed Genetic Algorithm for University Course and Student Timetabling Problems

Suyanto

Faculty of Informatics - IT Telkom, Jl. Telekomunikasi No. 1 Terusan Buah Batu
Bandung, West Java, Indonesia
suy@ittelkom.ac.id

Abstract. This paper describes an Informed Genetic Algorithm (IGA), a genetic algorithm using greedy initialization and directed mutation, to solve a practical university course and student timetabling problem. A greedy method creates some feasible solutions, where all specified hard constraints are not broken, as initial population. A directed mutation scheme is used to reduce violations regarding all given soft constraints and to keep the solutions feasible. Here, IGA creates a timetable in two stages. Firstly, IGA evolves a course timetable using any constraints regarding lecturer, class and room. This stage produce best-so-far timetable. Secondly, using some certain rules IGA evolves the best-so-far timetable using all constraints. The batch student sectioning is done by allowing the first stage timetable to change. Computer simulation to a highly constrained timetabling problem shows that the informed GA is capable of producing a reliable timetable.

1 Introduction

University timetabling problems are frequently categorized as course timetabling or exam timetabling. The major differences between both categories are [4]: exams must be scheduled so that no student has more than one exam at a time, but course must usually be scheduled before student enrollments are known; Exams often share rooms or are split across rooms, but only one course may be held in a room at any time.

However, those problems and the approaches vary in real life [2],[3]. For instance, course can be timetabled after student pre-enrollments, and then the students are automatically batch sectioned, and finally they could change their class using on-line re-sectioning [1],[3]. This approach is effective to reduce the student conflicts in very complex course timetabling problems. But, as the batch sectioning is performed separately after developing course timetable, some student conflicts may not be released. In this research, a new approach is proposed to do the batch sectioning by allowing the resulting timetable to change.

Researchers proposed many different techniques, but [2],[5],[6],[7] verify that genetic algorithm (GA) is powerful for very complex timetabling problems. This research focuses on an informed genetic algorithm (IGA). It is a GA using greedy initialization and directed mutation. The greedy method creates some feasible

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 229–236, 2010.
© Springer-Verlag Berlin Heidelberg 2010

solutions, where all specified fundamental hard constraints are not broken, as initial population. The directed mutation, inspired by violated directed mutation in [6],[7], functions to reduce violations regarding all given soft constraints and to keep the solutions feasible. Firstly, IGA evolves a course timetable using any constraints regarding lecturer and class. Secondly, using some certain rules IGA evolves an enhanced course timetable using all constraints regarding lecturer, class and students.

2 University Course Timetabling Problem

The complexity of university course timetabling problem (UCTP) varies depend on the size of university, the diversity of the course offerings and the specified constraints. In common UCTP, the constraints usually drive the complexity since so many people with different interests will be affected by the resulting timetable. Hence, a small university with not varying course offerings, but has so high constraints, is considered as a very complex UCTP.

The constraints are divided into two categories: hard and soft. Hard constraints are any fundamental and unbreakable constraints, such as no person may attend more than one meeting at any time, some special lecturers should be scheduled in their time constraints. Soft constraints are less important than hard constraints and usually impossible to avoid breaking for many cases. However, several universities have different manners to classify the constraints into hard and soft.

In this research, a real world data from a private university in Indonesia is used as a case study. The university has 5,349 students in 4 departments and 9 study programs. In one semester, there are 1,034 lecture meetings and 58,660 student meetings to schedule. It has 55 rooms categorized in 4 different capacities: extra large (XL), large (L), medium (M) and small (S). The complete characteristic of the data is illustrated by table 1. Since there are 24 time slots per week, both room XL and L have high occupancy of more than 85%. The most challenge in this case is that the courses are conducted in around 4 multiple classes in average and up to 27 multiple classes in maximum. It makes reducing student conflicts will be very complex.

Table 1. The characteristic of the real world data

Type of room	XL	L	M	S
Number of room	1	43	7	4
Room capacity	101-300	61-100	31-60	1-30
Number of lecture event	21	886	103	24
Room occupancy (%)	87.50	85.85	61.31	25.00

There are 11 constraints: 5 hard and 6 soft. The first three hard constraints are usual in common UCTP. But, two others are special in this case: all lecturers should be scheduled in time constraints defined by their departments; and special lecturers, those from other universities (external lecturers) and internal lecturers with high position, have privileges to define their time constraints. There are more than 30% special lecturers in the university. This is another challenge that makes the problem more complex. The hard and soft constraints are as follow:

Hard Constraints: HC1. No lecturer conflict; HC2. No class conflict; HC3. Any lecture meeting should be scheduled in the suitable capacity room; HC4. Lecturers should be scheduled in time constraints defined by their departments; and HC5. Some special lecturers should be scheduled in their time constraints.

Soft Constraints: SC1. Lecturer meeting spread; SC2. Class meeting spread; SC3. Some lecturers are better to be scheduled in their time preferences; SC4. Time constraints between meetings of the same lectures; SC5. Time constraints between different lecture meetings in the same group; and SC6. Minimizing student conflicts

3 Informed Genetic Algorithm

IGA generates a timetable in two stages as illustrated by table 2. Firstly, IGA evolves a course timetable using any constraints regarding lecturer, class and room. Secondly, using some certain rules IGA evolves an enhanced course timetable using all constraints.

Table 2. The IGA scheme

Stage 1
1. Greedy Initialization: N individuals
2. Directed mutation to produce N offspring
3. Calculate Fitness excluding student constraints
4. Select N best individual from $2N$ individuals
5. Back to step 2 until meet the stopping criteria
6. Return a best-so-far individual
Stage 2
1. Initialization: single best individual from stage 1
2. Directed mutation to produce one offspring
3. Calculate Fitness using all constraints
4. Select one best individual
5. Back to step 2 until meet the stopping criteria
6. Return the best individual

In the first stage, N individuals as initial feasible solutions are generated by greedy method. Afterward, each individual is selected as parent and then mutated using directed mutation so that produce N offspring. Then, N of 2 N

individuals are selected to live in the next generation based on their fitness which calculated using any constraint related to lecturer, class and room. Those processes are repeated until meet the specified stopping criteria. This stage produces a best-so-far individual as temporary timetable. Then, in the second stage, the best-so-far individual is evolved by IGA using some certain rules and the fitness is calculated by considering the all constraints.

3.1 Chromosome Representation

A chromosome is simply represented by an array of integer encoding time-room slot. Each gene represents an event. In the given data, there are 1,034 events to timetable. Hence, a chromosome has 1,034 genes and the alleles range between 1 and 1,320 since there are 55 available rooms and 24 time slots per week.

3.2 Fitness Function

The IGA is designed to minimize the violation score, i.e. the number of violation multiply by the defined penalty. The fitness function is formulated as

$$f = T * \sum_i^n p_i - \sum_i^n c_i * p_i \tag{1}$$

where T is the total number of events, c_i and p_i represent total violation of i-th constraint and its penalty respectively. A timetable with violation of all constraints will have $f = 0$. But, a timetable without violation will have maximum fitness.

The fitness concerns on soft constraints only since any hard constraint are designed to not violate. The penalty setting for the constraints is illustrated by table 3. This setting is designed to be proportional to effects of each constraint violation. For example, the penalty for a violation to lecturer meeting spread is 50 and for a student conflict is 1 since one class consists of around 50 students in average.

Table 3. The penalty setting for soft constraints

Constraint	Penalty
SC1	50
SC2	50
SC3	10
SC4	10
SC5	5
SC6	1

3.3 Greedy Initialization

For many problems, an initial population in GA is usually generated randomly. But, in common UCTP cases, the initial population is generated using some special knowledge to produce individuals representing feasible solutions. Here, a greedy method is used to develop feasible timetable candidates.

The method selects lecturers, based on the ascending ordered ratio of the number of time slot provided by them and the number of their meetings, to be scheduled on their time constraints. The ratio ranges between 1 and 24. A lecturer with ratio of 1 should be scheduled first since there is no spare time slot provided by the lecturer. If the method failed to produce a feasible solution candidate, it is repeated until a specified maximum trial.

3.4 Directed Mutation

Mutation operators in both stages of the IGA are designed to keep the timetable feasible and to reduce the soft constraint violations. They are implemented by directed mutations in slightly different schemes as illustrated by table 4.

In the first stage, each violating gene, i.e. gene containing any violation excluding student conflicts, is randomly mutated by changing the allele into another time-room slot which not breaks any hard constraint. Mutating a violating gene may release some other violating genes. Hence, mutated individuals will possibly contain lower violation. Some additional not-violating genes randomly selected to mutate in order to increase the diversity of the next population.

In the second stage, each gene containing any violation including student conflicts is mutated by changing the allele into other time-room slot which not breaks any hard constraint. This mutation focuses on reducing student conflicts by batch sectioning, but other soft constraints are also considered. In this stage, no additional genes randomly selected to mutate since the number of violating genes are usually much more than that of in the first stage.

3.5 Student Sectioning

The second stage of IGA focuses on reducing student conflicts by batch student sectioning. In the beginning, all students are randomly sectioned. Then, the batch sectioning is evolved using the second stage IGA. List all genes containing any violation including student conflicts. Afterward, mutate each violating gene using three successively ways:

1. Mutate each gene (lecture meeting) into another time-room slot which not breaks any hard constraint and reduce the violation score. This is a new additional way proposed to do the sectioning that allows the first stage resulting timetable to change.
2. For each student conflict in the time-room slot, change his/her class into another which releases the conflict.
3. For each student having any conflict in the time-room slot, swap his/her class with another student's class so that reduces their collective conflicts.

Table 4. The directed mutation for the IGA stage 1 and stage 2

```
Directed Mutation for the IGA stage 1
1. Permutation random $V$ violating genes excluding student conflicts
2. Random $M$ additional not-violating genes
3. For i=1 to $(M+V)$ do
      Mutate each gene into another time-room slot which not breaks any hard constraint
   End
```

```
Directed Mutation for the IGA stage 2
1. Find $N$ violating genes including student conflicts
2. For i=1 to $N$ do
      Mutate each gene into another time-room slot which not breaks any hard constraint
      If the mutated gen reduce the violation including student conflicts Then
         Do the mutation
      Else
         Cancel the mutation
      End
   End
```

4 Results and Discussion

A developed timetabling system was tested using a data explained in section 2. The testing focuses on the performance of the proposed directed mutation and batch student sectioning.

The system was examined using different number of additional genes, each 10 runs, to produce course timetables. The resulting timetables may have some violations since so many given soft constraints. The average fitness and violation for soft constraints SC1 to SC5 are listed in table 5. For both SC3 and SC5, the average violations are same for any additional genes. This result shows that directed mutation is quite powerful since it just violates 3 lecturer preference time slots from more than 300 given preferences, and there is no violation from 10 specified time constraints between different lecture meetings in the same group. For both SC1 and SC4, the average violations slightly fluctuate for any additional genes. Additional genes of 5 give higher violations than others, but it gives much lower violation for SC2. Hence, the additional genes of 5 give the highest fitness.

Table 5. The effect of randomly additional genes to mutate

Additional genes	Average fitness	SC1 Violation	SC2 Violation	SC3 Violation	SC4 Violation	SC5 Violation
5	127,519	17.3	2.4	3	71.6	0
10	127,398	17.5	4.7	3	71.2	0
20	127,385	17.5	5.0	3	71.0	0
30	127,396	16.9	5.3	3	71.4	0
40	127,416	16.9	5.1	3	70.4	0

Table 6. The advantage of additional way to student sectioning

Sectioning approach	Without additional way	With additional way
Student conflicts	809	741
Percentage (of 58,660 student meetings)	1.38	1.26
Occurrences of the additional way	0	38

The student sectioning procedure was tested by two different runs, with and without new additional way, to see the effect of the new way. Simulation results are illustrated by table 6. The additional way can reduce around 8.40% of the student conflicts. This result is produced after evolving the second stage IGA for 7 generations. In the first generation, where all students are randomly sectioned, the student conflicts are 14,665. Thus, the results show that the student sectioning with additional way is powerful to reduce the student conflict.

5 Conclusion

The IGA-based timetabling system is capable of producing a reliable course timetable and student sectioning. This result is influenced by directed mutation and new additional way in the student sectioning approach. For the given data, the additional way can reduce around 8.40% of the student conflicts.

Acknowledgments. Many thanks to IT Telkom for the financial support, and also to Rendy Munadi, Suwandi, Muhammad Chafidh Muttaqin, Hendra Winata, Muhammad Rifa'i S.H.A., Arinto Hardono, Erfan Fiddin, and all colleagues at IT Telkom for the kind and useful suggestions.

References

[1] Müller, T., Murray, K.: Comprehensive Approach to Student Sectioning. In: The 7th International Conference on the Practice and Theory of Automated Timetabling (2008)

[2] Nuntasen, N., Innet, S.: A Novel Approach of Genetic Algorithm for Solving University Timetabling Problems: a case study of Thai Universities. In: Proceedings of the 6th WSEAS International Conference on System Science and Simulation in Engineering (2007)

[3] Murray, K., Müller, T., Rudová, H.: Modeling and Solution of a Complex University Course Timetabling Problem. In: The 6th International Conference on the Practice and Theory of Automated Timetabling (2007)

[4] Burke, E.K., Jackson, K., Kingston, J.H., Weare, R.: Automated University Timetabling: The State of the Art. The Computer Journal 40(9), 565–571 (1997)

[5] Burke, E.K., Elliman, D., Weare, R.F.: A Hybrid Genetic Algorithm for Highly Constrained Timetabling Problems. In: Proceedings of the 6th International Conference on Genetic Algorithms, pp. 605–610 (1995)

[6] Ross, P., Corne, D., Fang, H.-L.: Improving Evolutionary Timetabling with Delta Evaluation and Directed Mutation. In: Davidor, Y., Männer, R., Schwefel, H.-P. (eds.) PPSN 1994. LNCS, vol. 866, pp. 556–565. Springer, Heidelberg (1994)

[7] Corne, D., Ross, P., Fang, H.-L.: Fast Practical Evolutionary Timetabling. In: Fogarty, T.C. (ed.) AISB-WS 1994. LNCS, vol. 865, pp. 251–263. Springer, Heidelberg (1994)

Part III

Agent Systems, Robotics and Control

Evaluation of a Communication Platform for Safety Critical Robotics

Frederico M. Cunha, Rodrigo A.M. Braga, and Luis P. Reis

Artificial Intelligence and Computer Science Laboratory - LIACC,
Faculty of Engineering of University of Porto - FEUP,
Rua Dr. Roberto Frias s/n, Porto, Portugal
fredericom.cunha@gmail.com, rodrigo.braga@fe.up.pt, lpreis@fe.up.pt

Abstract. As the number of handicapped people increases worldwide, Intelligent Wheelchairs (IW) are becoming the solution to enable a higher degree of independence for wheelchair users. In addition, IW Projects relevance is increasing, mainly in the fields of robotics and safety-related systems due to their inherent and still unresolved problems related with environment uncertainty, safe communications and collaboration methodologies. This paper describes the development of new communication system, based on multi-agent systems (MAS) methodologies and motivated by Intelligent Wheelchair systems, as a mean to enable fault-tolerant communications in open transmission systems and as an agent collaboration enabler. It provides an overview of the related work, the background and the main constraints to system development. It proposes and discusses a new communication model, based on messages, for multi-agent systems, that tackles the problems that exist in a dynamic and uncertain environment in the field of mobile robotics. The achieved results enable us to conclude on the effectiveness of the proposed comunication model and its adequacy to the field of mobile robots in dynamic environments while establishing a comparison with a commonly used Multi-Agent Platform, JADE.

1 Introduction

The World Health Organization (WHO), estimates that around 2% of world population (130 million people) live with physical handicaps. This increasing number is due to several physical disabilities caused by congenital deficiencies, diseases, accidents, wars, and the aging of population as consequence of the increase in life expectancy, [1].

The most common aid for this kind of mobility problem is the wheelchair, specifically the electric wheelchair. However, it does not provide the autonomous life that most users want. To address this problem, numerous Intelligent Whellchair (IW) related projects have been announced, and are under development in the last years. The increase study of this problem, led to a globally accepted view of the main functional requirements for such systems. According to [8] and further developed by, [2], the main functions of an IW, can be categorized as the following:

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 239–246, 2010.
© Springer-Verlag Berlin Heidelberg 2010

- **Interaction with the user** this includes hand based control(such as joystick, keyboard, mouse, touch screen), voice based control, vision based control and other sensor based control;
- **Autonomous navigation** this must provide safety, flexibility and robust obstacle avoidance;
- **Communication systems** to enable and provide interaction with other devices, such as other IWs, automatic doors, remotely operated control software for medical staff.

Although many IW projects exist, the majority tends to concentrate their efforts in the interface with the user or in the navigational system. The communication system is rarely described in these projects and scarcely treated as an important and vital piece of an Intelligent Wheelchair. A common solution seen for the communication system is the use of CORBA based systems, or other technologies that enable communication through object sharing techniques, as seen in [10]. These allow an easier and fast development but are rather limited when addressing the problems of mobile robotics.

The SENA robot, presented in [6], is one of the few IW projects that approaches the communication system. One of the first references to it was the proposal of an architecture for this type of system, ACHRIN. This system although started as a rigid communication platform, it latter evolved into a Multi-agent Systems (MAS) based communications, [5], thus taking advantage of MAS maturity, robustness, scalability and easy management of information and entities.

This work's main objectives are: to identify in the field of mobile robotics, with emphasis on IW cooperative communication, the main constraints to a safe and secure communication system, propose and test the solutions that can address the found constraints and compare the test results to previously used methods in a adequate environment.

The rest of this article is divided into 4 additional sections. Section 2 provides an overview of the IntellWheels project and its different modules as the motivation for this study. Section 3 provides a description of the applicable constrains, linking them to the described project and to the field of mobile robotics. A detailed description of the proposed solutions is also given in this section. Section 4 describes the test environment, the applied test methodology and presents the test results. Section 5 discusses the relevance of the test results and the applicability of the proposed methods to the field of mobile robots and robots in dynamic and safety critical environments.

2 Intellwheels Project

The IntellWheels' main objective is to create a complete intelligent system, hardware and software, which can be easily integrated into any commercially available electric wheelchair with minor modifications to its structure. It aims to contribute to the advance in this field by providing solutions for some of the problems. IntellWheels' target user ranges from the wheelchair users with small locomotion disabilities to those with severe handicaps.

With the collaboration of different persons and integrating their work in the project, the architecture evolved along the time, from a static and rigid control layer architecture. The current architecture is composed of several logical and coherent modules organized as different micro-agents within a macro agent, the Intelligent Wheelchair, and can be seen in Fig. 1.

While these software modules are parts of the wheelchair agent, they are themselves, autonomous agents demanding data from and serving data to other software agents. This modular organization, enables a higher degree of upgradability for the architecture, and opens new paths for collaboration between IW. An example is the possibility for an IW to share or run a planning service for another IW, or for an IW agent to share its control interface with another IW.

The new architecture, its organization and the environment constraints, served as motivation for a new communication system, described in this article. This new system was designed to enable the easy development and integration of new functional modules as also to facilitate the use of new cooperative and communication methodologies while maintaining a high degree of reliability. Thus enabling the transmission of messages, the dynamic configuration of the different agents and their rapid adaptation to a new communication environment.

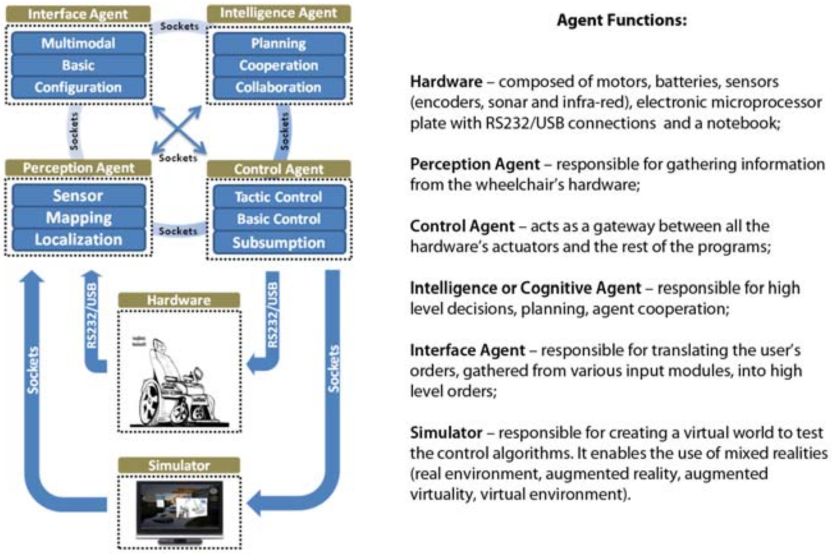
3 System Description

Normally, a multi-agent platform, such as Jade, would be used to enable communications as well as organize and manage the different agents. However with common multi-agent platforms it is not possible to customize or enhance system functionality in order to adapt the systen to a specific reality.

In Intellwhells case, the solution was to develop a new communication system, based on MAS methodologies, that could address the following constraints: multiple agent support, compatibility with other communication languages and communication systems, an easy to use and to configure systems interface, system's reconfiguration according to changes in the physical and networked envirounment, aplicable to open transmission systems and an easy to use interface to facilitate knowledge sharing amongts agents.

Taking these constraints into account, and the unique characteristic that all software for Intellwheels behaves as individual agents within the wheelchair, the new system was developed as two separate platforms. The first, a global platform, applied to all software within the same network environment and based on the FIPA standard, [4]. The second, a local platform, responsible for the management of all local services needed to run an IW.

Central in this architecture is the election of a Container entity and the distribution of a Local Agents List (LAL), as well as a Global Agent List (GAL), while using message-oriented paradigm as opposed to object-oriented. The use of object sharing platforms was discarded due to their higher architecture complexity, firewall traversal and dependibility on proprietary extensions. In these lists are the applications' configurations that enable communications (distribution of the public encryption key) between agents.



Agent Functions:

Hardware – composed of motors, batteries, sensors (encoders, sonar and infra-red), electronic microprocessor plate with RS232/USB connections and a notebook;

Perception Agent – responsible for gathering information from the wheelchair's hardware;

Control Agent – acts as a gateway between all the hardware's actuators and the rest of the programs;

Intelligence or Cognitive Agent – responsible for high level decisions, planning, agent cooperation;

Interface Agent – responsible for translating the user's orders, gathered from various input modules, into high level orders;

Simulator – responsible for creating a virtual world to test the control algorithms. It enables the use of mixed realities (real environment, augmented reality, augmented virtuality, virtual environment).

Fig. 1. Current Intellwheels Architecture diagram and description of the different agents

The Container, named after the Jade Container entity, [7], was designed to be responsible for the LAL's maintenance. These maintenance operations include the following: creation, update, deletion. However and contrary to the Jade system, the Container was not designed as a separate entity or as the base for agents creation and their activity. The idea behind this, is that it is admissible and probabal for a whellchair to lose network connectivity or to change its network configuration but it is no acceptable for these changes to cause a system's malfunction.

Common to both architectures is the use of the FIPA-ACL communicative acts for the messages' structure, as well as FIPA-SL for language representation, thus providing a strong base for compatibility with other objects or systems.

3.1 Micro Architecture, the Local Platform

As above mentioned, the local platform is organized and maintained by a Container entity. It is not however, a separate software. The system was designed for the container algorithms to remain inside the communication's structure, thus making it a part of all applications. This way it is also possible to start an agent and for it not to depend directly on the communication system's configuration, state machine and resources to performe its function.

The system's architecture was designed as five separate layers with their respective receiving and sending handling methods and interfaces, as seen in Fig. 2 a), running in parallel. This way, it becomes possible for the user to choose which layers should be applied to the application, without compromising the agent's functionality while following the OSI Reference Model and implementing fault tolerant methods described in [9] and [3].

Another of the features of this structure is the full customization of the specific messages' handling functions to be applied within each layer, giving the user total control over the resulting message envelope, seen in Fig. 2 b).

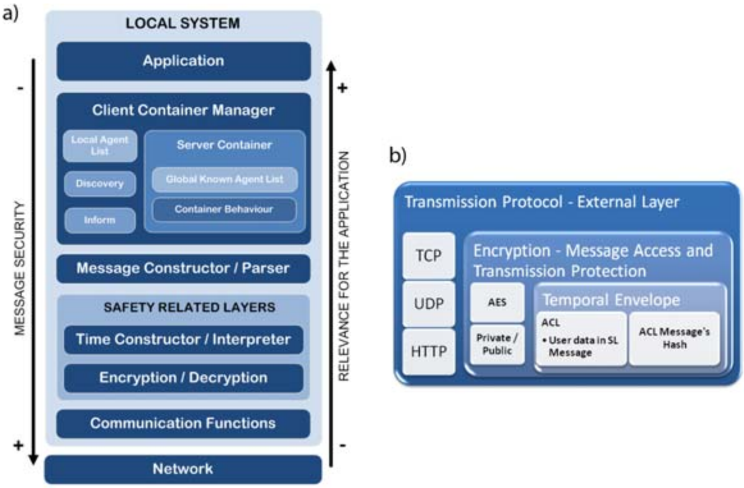


Fig. 2. Local platform structure, a). The software is organized into 5 independent layers that can be used to construct the application's communication system. The resulting default message envelope can be seen in b).

The layer's functions are the following:

- **Communications layer** it is responsible for receiving and sending messages from and to the message transport layer. Enables the user to choose between TCP/IP, UDP or even HTTP messages. This layer also prevents the interpretation of repeated messages present in the physical media and enables the retransmission of messages thus preventing packet loss in the media layer. It also prevents the application from receiving messages with a size larger than the one specified by the user. When not specified the default messages' sizes are 8kBytes for UDP and 16kBytes for TCP;
- **Decryption and Encryption layer** responsible for the message's security, preventing the interception and modification of messages. The Encryption method is chosen accordingly to the message's destination and the knowledge requested, in the case of platform messages exchange. The user's message encryption can be chosen by the user, and treated accordingly. The possible encryption methods involve the use of a private and public key pair or an AES pre-shared key. It also performs message integrity checking by cross-referencing the message with the transmitted message's hash;
- **Time layer** responsible for adding a time stamp to the message to send, for organizing the received messages according to the time stamp and for the detection and

elimination of injected packets. Another function that it performs is the configuration and synchronization of the local system's clock with a networked NTP clock, if available. This configuration is done automatically and only if the application is the local Container;
- **Message Constructor / Parser layer** responsible for the construction of the message according to the FIPA-ACL standard and represented using FIPA-SL. It is also responsible for selecting the messages that should be accepted by the application according to their correct structure configuration and to the sender's presence in the platform, thus stopping any communication from an outside application;
- **Client Container Manager** responsible for managing the application's organization and integration in the local and networked platforms, implementing methods like replication, fault detection, recovery and discovery. It also implements the user interface with the communication system by enabling direct access to the Local Agent List and Global Agent List.

An additional platform component was designed, the Log. Linked to all of the previously presented layers by a buffer, it is responsible for logging the activity that occurs within the application, either messages sent by the user or by the platform's maintenance functions. It's interface enables the user to choose the layers to be logged as well as the functions that apply to the log. By default every line presented in the log file is identifed by a timestamp and by a custom message to identify the function that generated it. However when encryption is enabled, no log is performed to the encrypted messages as a security measure, to prevent encription key discovery. It's also possible for the user to use the Log component in the agent functions to record their state.

4 Tests: Scenario, Protocol and Results

This section presents the testbed, the test scenarios and the achieved results.

In order to validate the proposed solutions, the above described architecture, was implemented in Pascal, using Borland Delphi Professional v7.2 IDE. Jade v3.7 and the Eclipse Platform v3.4.2 were used to develop and test the Jade agents described. Also, as an support for the testing, an automated application laucher and data logger, was developed. For all tests the application Process Explorer from Sysinternals, was used as a resource monitor.

All tests were repeated 20 times with the same conditions and all data was analysed and represented with a confidence interval of 95%.

To validate the local platform two test scenarios were established, each with different objectives.

The first test's objective was to measure the performance, effectiveness and scalibity of the communication platform. It consisted on a agent sending a message, with a fixed size of 500 Bytes, that would be redistributed to all agents in a serialized manner. The time that the message took to be passed between all agents and return to the initiator, was measured and analysed. To gather this test's data, one agent was implemented to follow a protocol, that can be seen in Fig. 3 a), when instantiated more than once. In this picture, x represents a integer number used as a control mechanism for message differentiation, and

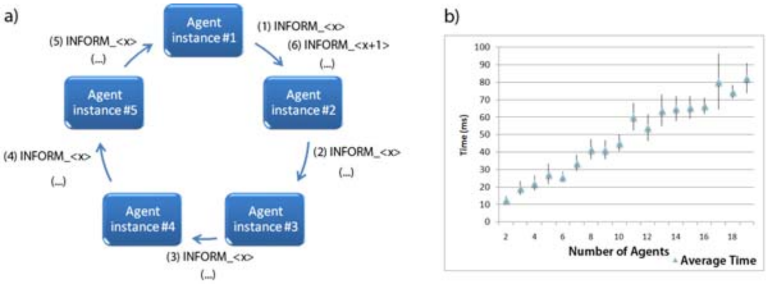


Fig. 3. Implemented protocol in the first test's scenario, a). Test's results, b).

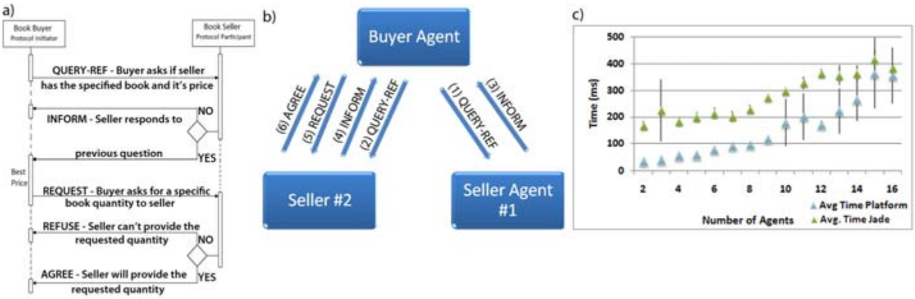


Fig. 4. Book buyer and book seller agents' protocol, a). An example of message exchange in this test scenario with seller #2 providing the book's best price, b). Test's results for Jade and the described platorm, c).

the numbers in between round brackets the interaction sequence. This test was repeated using the maximum number of 19 agent instantiations. The gathered results not only demonstrate that the platform was able to deliver all messages in the correct order without errors, but also that the platform scales linearly with the amount of agents present.

The second test's scenario objective was to evaluate the performance and scalability of the platform in a well known agent simulation case, and to compare the results with the ones obtained, in the same scenario, using the Jade platform. The implemented scenario followed the Book Seller Book Buyer protocol, seen in Fig. 4 a) and b), for the example of one buyer and two sellers. For the test, only one buyer agent was instantiated, while the number of seller agents increased in each test round to the maximum of 15 agents. The time needed by the buyer agent to successfully accomplish the book's request and acquisition, was logged for both the Jade platform and the proposed architecture. The results in Fig. 4 c), allow a comparison to be established between the two platforms. While Jade is slower with a smaller number of agents, it's test's results are closer together, with small variations between them. On the other side, the proposed platform is able to communicate very fast with a small number of agents, but when this number increases, the result's interval increases also becoming wider.

5 Conclusions

With the rapid and growing number of Intelligent Wheelchairs under development, more attention must be paid to the development of communication systems. In this field, object interaction is absolutely needed, in order to enable cooperation between objects. This paper's main objective was to propose and describe new solutions for agents' communication, applicable to the field of mobile robots in dynamic environments. The tests results achieved using the proposed architecture, show that the platform can in fact provide cooperative communications between agents, and provide a new solution for agent-based systems. Moreover, the defensive methods implemented, proved to be able to protect data transmission and system functionality. Although the experiments show that there is still work possible to make the platform more scalable, the comparison with JADE shows that for a number of agents smaller than 12, the average time achieved by the proposed platform is less than half of Jade's time.

References

1. Braga, R.A.M., Petry, M., Moreira, A.P., Reis, L.P.: Intellwheels: A Development PlatForm for Intelligent Wheelchairs for Disabled People. In: Proc. of the 5th Int. Conf. on Inf. in Control, Aut. and Robotics, May 2008, vol. I, pp. 115–121 (2008)
2. Braga, R.A.M., Petry, M., Moreira, A.P., Reis, L.P.: Concept and Design of the Intellwheels Platform for Developing Intelligent Wheelchairs. Informatics in Control, Aut. and Robotics 37, 191–203 (2009)
3. EN 50159-2. Railway applications – Communication, signaling and processing systems, Part 2: Safety related communication in open transmission systems. Europ. Commitee for Electr. Standardization (March 2001)
4. FIPA, Found. for Intelligent Physical Agents (October 2009), http://www.fipa.org
5. Galindo, C., Cruz-Martin, A., Blanco, J.L., Fernández-Madrigal, J.A., Gonzalez, J.: A Multi-Agent Control Architecture for a Robotic Wheelchair. In: Applied Bionics and Biomechanics, vol. III, pp. 179–189 (2006b)
6. Galindo, C., Gonzalez, J., Fernandez-Madrigal, J.A.: A Control Architecture for Human-Robot Integration. Application to a Robotic Wheelchair. IEEE Trans Systems, Man Cybernet-Part B 36 (2006)
7. JADE, Java Agent Dev. Framework (October 2009), http://jade.tilab.com
8. Jia, P., Hu, H.: Head Gesture based Control of an Intelligent Wheelchair. In: CACSUK-11th Ann. Conf. Chinese Aut. Comp. Soc. in the UK, Sheffield, UK (2005)
9. Malm, T., Hérard, J., Boegh, J., Kivipuro, M.: Validation of Safety-Related Wireless Machine Control Systems, NT Tech. Report, Nordic Innov. Centre, Norway (2007)
10. Prenzel, O., Feuser, J., Graser, A.: Rehabilitation robot in intelligent home environment - software architecture and implementation of a distributed system. In: 9th Int. Conf. on Rehab. Robotics, ICORR 2005, June 28-July 1, pp. 530–535 (2005)

How to Gain Emotional Rewards during Human-Robot Interaction Using Music? Formulation and Propositions

Thi-Hai-Ha Dang, Guillaume Hutzler, and Philippe Hoppenot

Laboratory IBISC, University of Evry Val d'Essonne, Boulevard Francois Mitterrand
91025 Evry, France
{hadang,guillaume.hutzler,philippe.hoppenot}@ibisc.univ-evry.fr

Abstract. In this paper, we present arguments for the need of emotion modelling and we define elements for a study in Human-Robot Interaction (HRI) using music. We also propose an adaptation of our generic model of emotions (GRACE) to give a precise idea of how to design emotional intelligence for a robot with music-related abilities.

1 Introduction

Thanks to the rapid development of artificial intelligence (including pattern recognition, speech processing, motion planning, machine learning, etc.), robots nowadays have gained various abilities to do repetitive, meticulous, dangerous tasks with high precision at great speed. Robots are thus greatly appreciated for replacing human in everyday/industrial/outer space tasks.

Besides, robotic applications are also present in assistive tasks, where comfortable experience is in the first rank of exigencies. This entails demands on soft skills for robots in order to deal with human mental states, to mimic human social interaction and to gain cooperation of human. Experimentations of robotic applications have also shown the emotional rewards to human in interacting with intelligent robots.

Moreover, various researches interested in music and emotions have claimed that the emotional reward of music remains an important reason for the universal appeal of music [2,4]. So, can music be involved in human-robot interaction to gain emotional rewards to human? This paper is an attempt to formulate such a case study.

In section two, we highlight the main characteristics of HRI to be taken into account in robotic application design. We present also in this section the findings of emotional rewards during human-robot interaction and recent works on emotion modelling in computer science. In section three, the first subsection lists important elements to be focused on in studying the role of emotional rewards in HRI using music. In the second subsection, we propose an adaptation of GRACE, our generic model of emotions, in order to take into account these elements. Finally, in section four, we give some perspectives for our study on emotional rewards in HRI using music.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 247–255, 2010.
© Springer-Verlag Berlin Heidelberg 2010

2 The State of the Art

Robotics, since the use of diverse robotic applications in human life, has been opening large perspectives of multi-disciplinary works, e.g. machinery design, control mechanism, beautiful appearance, intelligent reaction, good performance of task accomplishment and also efficient interface of interaction (i.e. human-robot interaction - HRI). This HRI itself is also a field of study where several researches have been focused. The number of research projects on communication channels between human and robot (keyboard, mouse, voice, speech, gesture, etc.) increases with time. Many of them have gained successes in improving communication efficiency. Soon afterward, importance of the emotional aspect of communication began to draw attention. Researchers in HRI started to study/apply knowledge in emotion theories into their applications. In this section, we will present advances in HRI research and the need of emotion modelling in robotic applications.

2.1 Human-Robot Interaction - Need of Emotion Modelling

Natural and affective interaction - this could be considered as the principle goal of all works on human-machine interaction. Being a specific branch of research in human-machine interaction, research in HRI, specially HRI for robotic assistance, is focused on such a kind of interaction. Kiesler and Hinds, in the introduction of *Special Issues in Human-Robot Interaction* [10], mention some important points to be considered in HRI research, particularly in robotic assistance, which are:

– *Anthropomorphic model*: this relates to the need of using natural channels of communication to facilitate the interaction of humans with the robot. The more natural human-robot communication is, the better the robot gains cooperation from human. Several channels could be considered such as voice, speech, gesture, etc.
– *Physically close to other robots, people, objects*: usually, robotic assistance involves human participation in a close distance. This characteristic has to be seen as of major importance to study HRI. This importance appears in the necessity of robots having abilities to work in a dynamic and challenging environment. This interaction also entails the question of safety issues for both human and robot during interaction.
– *Sufficient knowledge of the context of use*: this is for the design of an effective interaction scheme to provide appropriate HRI applications that suit the social/ethnical standards of the user. This also includes the robot's abilities to automatically learn about themselves and their world to well adapt to changing situations.

Experimentations with intelligent robots have repeatedly demonstrated the importance of emotional rewards in robotic applications. Since 2003, several experiments with Paro [16], a seal-like robot, held in Japan, in United States of America, in France, etc., have shown that people (elderly, children) experienced some kind of comfortable feeling while interacting and living with robot Paro. In a survey of Forlizzi et al. (2004) [20], emotional rewards were mentioned many times when people talked about the capacity of assistive materials to permanently keep elderly in connection with others (their relatives, their friends, their neighbours, etc.).

Being encouraged from works of psychologists [1,2,3,5] on the importance of emotional reactions in the adaptation abilities of human, research in artificial intelligence has begun studying the emotional intelligence of human and trying to implement it in robots to render more natural behaviour and so to gain success in interaction with human.

2.2 Researches in Modelling Emotions for Computerized Applications

Advances in robotics and artificial intelligence have strongly changed the way people experience computerized applications. With the discovery of emotional rewards for human during interaction with his robotic equipments, research on modelling emotions has come to life. This research direction requires to be based on studies on emotion theories in psychology. Among several theories, some are often used as basis by almost all works. These theories are the theory of Ortony et al [8] on *event appraisal*, the theory of Lazarus [7] about *appraisal* and *coping*, and the theory of Scherer [3,6] about *emotional processes*.

In the aim of modelling and implementing emotions into computerized application, several approaches have been explored. El-Nasr et al.[11] explored the use of fuzzy logic to calculate emotional intensities. This model was implemented in a simulation named PETEEI - a Pet with Evolving Emotional Intelligence. Embodied Conversational Agents, such as ParleE (by Bui et al, 2002) [12], Greta (Pelachaud, 2003) [17], GALAAD (by C. Adam, 2005) [19], have been developed, taking into account the appraisal process of emotions and even personality traits. There is also a framework for training applications called Mission Rehearsal Exercise, developed by Gratch and Marcella in 2004 [18], that provides virtual scenarios for health intervention, marketing and entertainment. The emotional aspect of this proposition is the adaptation of *event appraisal* and *coping* (two important aspects of emotions, according to Lazarus [7]). Now come the applications in the domain of HRI. In 2002, C. Breazeal at MIT lab attempted to establish an interaction between a robot and a human inspired by the relation between a baby and his/her parents [14]. Emotions experimented are anger, distaste, fear, sadness, and happinesss. Recently, MIT lab has also announced their advance in modelling emotions into another robot named Nexi - a Mobile Dexterous Social Robot with more human-centric communication and interaction abilities [15]. In 2008, we proposed a model of emotions for robotic applications, called GRACE [21,22]. GRACE is based on the psychological theories that we mentioned previously. This model is considered generic as it can instantiate anterior models of emotions in the domain of computer science by using some of its components. A comparison of these models of emotions can be found in [21].

3 Proposition for the Study on Emotional Rewards of HRI Using Music

The universal appeal of music mostly comes from the emotional reward that music offers to its listeners. According to Zentner in [4], emotional impacts of music explain its prominent role in people's everyday lives. In [2], music is ascertained to have direct effects to treatment of emotion disorders. Now imagine that a robot can detect the emotional state of its partner when he listens to the music (or he plays the music) and then react in an appropriate manner, can this robot enhance emotional rewards for its partner during/after this interaction? This is the question we try to answer in our study.

To this end, beside the study of emotional impacts of music to human, we need a model of emotions to implement the robot's behaviour so as to enable an emotionally rich interaction with humans. With GRACE, we already described and modelled the emotional process for robots. However, the use of music during human-robot interaction requires some specific adaptations. In this section we study these requirements and propose modifications to suit the case of using music.

3.1 Important Elements in Studying HRI Using Music

Given that the objective of our work is to study the mental rewards during human-robot interaction using music, the interaction scenario will be between a musician and a robot with music-related abilities. Thus, the desired robot should be equipped with a cognitive process that helps it to perceive its partner's emotions via his music (and maybe his visible behaviour) and then to react in an appropriate way. Important information used for the robot's intelligent behaviour in this study should be:

- The emotional interpretation of the music played by a musician: this represents the interpretation done by the musician. The robot has to know how its partner feels via his music. As claimed in [4,5], it is feasible to construct a learning module that takes a piece of music as input and reproduces the emotions probably experienced by a specific individual.
- The robot's mental state (specifically personality trait of robot: introversive, extroversive, aggressive, curious, etc.). This mental state has an important impact on the selection of action alternatives. So, with the same emotional state of musician (interpreted by the module of interpretation), the robot can change its preferred tendencies of action based on its own mental state. For example, if the detected emotion of the musician is sadness, then the robot, if it is extroversive, can have action tendencies to show empathy to its partner; but if the robot is aggressive, it could have tendency to disturb the musician (like making noises, performing exciting movements, etc.).
- The robot's current goals (excite its partner, get rid of him, disturbing ongoing situation, etc.). This current goal gives hints to select preferred tendencies. When the robot wants to excite its partner, it should prefer actions that

decrease the negative emotions of its partner and increase positive impact (such as performing funny dances, showing empathy by dancing coherently with the music played by its partner).

- Ability to guess future actions of its partner (the musician). This ability allows the robot to predict upcoming events to better adapt its behaviour. This could be done by deploying a prediction module based on a predefined scenario or applying pattern recognition on the set of events over time.

To take into account important elements presented above, the model GRACE, which remains abstract for general use case, will be adjusted and concretized into an adaptation called MACE-GRACE (Music-adapted Architecture to Create Emotions - GRACE).

3.2 MACE-GRACE - Adaptation of GRACE in the Context of HRI Using Music

Being constructed by merging psychological theories on emotions and computational models of emotions, GRACE aims at simulating an entire emotional process of human. The emotional process of GRACE is described in Figure 1:

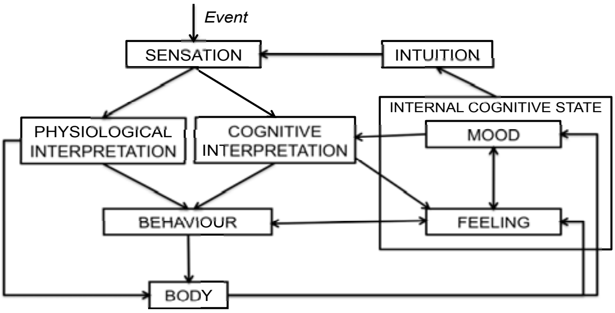


Fig. 1. Model GRACE - Generic Architecture to Create Emotions

For our study on emotional rewards in HRI application, some details must be precisely pointed out:

- *Sensation*: this GRACE module does the collection of information necessary for emotional processing. In our case, information collected will be musical stimuli (let's call them 'musical events') and environment information required for robot safety verification.
- *Physiological interpretation*: this GRACE module simulates the physiological symptoms of the emotional process (like temperature sensations, respiratory and cardiovascular accelerations and decelerations, trembling and muscle spasms) to show reflex reactions. In our case, only the safety of the human and the robot will be verified in this module to ensure a good performance of the robot; other physiological symptoms will not be taken into account.

- *Cognitive interpretation*: in GRACE, this module is in charge of analyzing the emotion-eliciting events to have a global view of current situation (more specifically, the emotion-related aspects of the current situation). Result of this analysis is used to decide action tendencies to respond to events. In our case, what we want in output of this module are the current emotions expressed in the musical events played by the robot's partner. We tend to use a learning module here to analyze the musician's emotions via his music. The result of this module can help the robot to choose appropriate actions to respond to the musical events.
- *Mood*: this component contains the robot's current mental state, its current stand faced to current situation, its current goals/needs in term of assigned tasks and even its habits(e.g. its preferred action tendencies, preferred situation/events). To focus on the goal of our study (mental rewards for humans during the interaction with a robot having music-related abilities), we will code in this module only the robot's goal/need, and a simplified mental state.
- *Feeling*: this module plays the role of memorization of the running process. It captures the reaction of all components to events and is used as a database for adaptation and memory recall abilities. In our study, this part would store the musician's attitude (history/personality of musician). This module can help the robot to decide its preferred strategy of action in regard to the attitude of musician.
- *Intuition*: this module of GRACE is supposed to do the anticipation of future events, to generate imaginary events (i.e. internal events). We propose in our study to relate it to some memory recalls or anticipations of future reaction of the musician so that the robot can react rapidly to new events.
- *Behaviour*: in GRACE, this module is for the selection of an action to respond to an event. This selection takes into account information of both the external situation and the robot's internal state. In our case study, this would be the selection/regulation of robot's movements (accelerate or decelerate the movement of the hands, turn the body around, make some noise, etc.) according to the musician's emotion expressed in his music.
- *Body*: this module executes the robot's motor expression in response to input events. Generally, this includes reflex reactions from *Physiological Interpretation* and reactions selected by *Behaviour* module. In our study, this part will execute the robot's music-related reactions to show that the robot understands the emotional expression in the playing music and shows its response in relation with the current situation and its own personality.

To take into account these issues, we propose MACE-GRACE, an adaptation of GRACE for HRI using music, described in Figure 2. The analysis process of robot's intelligence when it captures musical events will be as follows:

The physiological interpretation is in charge of verifying the physical impact of events to robot to ensure the robot's safety based on its position, its battery level, etc. The simulation of human-like physiological stimuli (such as body temperature, heart beat, respiration, or even reflex reactions) of the robot is not taken into account in this study. The robot's cognitive interpretation will

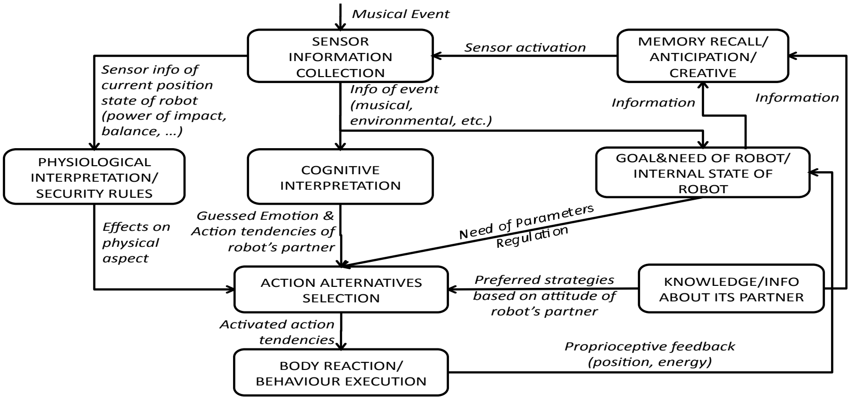

Musical Event
SENSOR INFORMATION COLLECTION
Sensor activation
MEMORY RECALL/ ANTICIPATION/ CREATIVE
Sensor info of current position state of robot (power of impact, balance, ...)
Info of event (musical, environmental, etc.)
Information
Information
PHYSIOLOGICAL INTERPRETATION/ SECURITY RULES
COGNITIVE INTERPRETATION
GOAL&NEED OF ROBOT/ INTERNAL STATE OF ROBOT
Effects on physical aspect
Guessed Emotion & Action tendencies of robot's partner
Need of Parameters Regulation
ACTION ALTERNATIVES SELECTION
Preferred strategies based on attitude of robot's partner
KNOWLEDGE/INFO ABOUT ITS PARTNER
Activated action tendencies
Proprioceptive feedback (position, energy)
BODY REACTION/ BEHAVIOUR EXECUTION

Fig. 2. Adaptation of GRACE in the context of HRI using music

measure its partner's emotions based on the music that he plays. This measured value will be sent to the action selection to determine an appropriate response to the current emotional state (measured by the cognitive interpretation). This phase of selection is also influenced by the robot's current goal/need (calm down the musician or excite the current ambiance, amplify the current state of the musician) and with regard to the musician's attitude. The execution of the selected action will be stored in the robot's database along with the behaviour of the musician at that moment. This memory allows the robot to make predictions of the future behaviour/music of its partner so that it could have an adaptive behaviour in the whole conversation.

With this adaptation, we tend to integrate in our robot an artificial emotional intelligence to interact with a musician. This emotional intelligence can help the robot not only perceive the emotional state of its partner but also react to diverse situations in an adaptive manner in term of emotion-based strategy.

4 Conclusion

Throughout the paper we have presented major characteristics of robotic applications and then mentioned the role of emotional rewards during human-robot interaction. We also proposed an evolution of our model of emotions previously proposed for computerized applications. This evolution is in the aim of studying the emotional rewards in human-robot interaction using music. We consider the study as an attempt to implement emotional intelligence into a concrete computerized application, more specifically, an assistive/entertaining robot. This could have interesting applications in such fields as music-based therapy, emotion-based movements, preferences of robot abilities, etc.

The realization of this adaptation is a long way to go. It consists in implementing the model, doing the experimentation with the robot and then analyzing the results. Furthermore, the implementation of the model could also concern works in database construction (including choices in music style, user preferences, music descriptors, definition of musical event, etc.), system specification, programming tasks, etc. Yet, we demonstrated in this study that GRACE, our generic model of emotions, was versatile enough to be easily adapted to the specific needs of emotional interaction using music.

References

1. Darwin, C.: The expression of the emotion in man and animals. Murray, London (1872); Reprinted: University of Chicago Press, Chicago (1965)
2. Gebhardt, S., von Georgi, R.: Music, mental disorder and emotional reception behavior. Music Therapy Today VIII(3) (2007)
3. Scherer, K.R.: Emotion, the psychological structure. In: Smelser, N.J., Baltes, P.B. (eds.) International Encyclopedia of the Social and behavioral Sciences. Pergamon, Oxford (2001)
4. Zentner, M., Grandjean, D., Scherer, K.R.: Emotions Evoked by the Sound of Music: Differentiation, Classification and Measurement. Emotion 8(4), 494–521 (2008)
5. Scherer, K.R.: Which emotions can be induced by music? What are the underlying mechanisms? And how can we measure them? Journal of New Music Research 33(3), 239–251 (2004)
6. Scherer, K.R.: On the nature and function of emotion: A component process approach. In: Scherer, K.R., Ekman, P. (eds.) Approaches to emotion, pp. 293–317. Erlbaum, Hillsdale (1984)
7. Lazarus, R.S.: Emotion and Adaptation. Oxford University Press, Oxford (1991)
8. Ortony, A., Clore, G.L., Collins, L.: The cognitive structure of emotions. Cambridge University Press, Cambridge (1988)
9. Myers, I.B.: Gifts Differing: Understanding Personality Type. Davies-Black Publishing (May 1, 1995); ISBN 0-89106-074-X (1980)
10. Special Issue of Human-Computer Interaction, vol. 19(1-2) (2004)
11. El-Nasr, M., Yen, J., Ioerger, T.: FLAME - A Fuzzy Logic Adaptive Model of Emotions. Journal of Automous Agents and Multi-agent Systems 3, 219–257 (2000)
12. Bui, T., Heylen, D., Poel, M., Nijholt, A.: ParleE: An Adaptive Plan Based Event Appraisal Model of Emotions. In: Jarke, M., Koehler, J., Lakemeyer, G. (eds.) KI 2002. LNCS (LNAI), vol. 2479, pp. 129–143. Springer, Heidelberg (2002)
13. Rousseau, D.: Personality in Computer Characters. In: Proceedings of the 1996 AAAI Workshop on Entertainment and AI / A-Life, pp. 38–43. AAAI Press, Portland (1996)
14. Breazeal, C.: Emotion and sociable humanoid robots. Hudlika, E. (ed.) International Journal of Human-Computer Studies 59, 119–155 (2003)
15. Project MDS: Personal Robots Group, MIT Media Lab, http://robotic.media.mit.edu/projects/robots/mds/overview/overview.html
16. Official site for Mental Commitment Robot (PARO), http://paro.jp/english/
17. de Rosis, F., Pelachaud, C., Poggi, I., Carofiglio, V., De Carolis, N.: From Gretas Mind to her Face: modelling the Dynamics of Affective States in a Conversational Embodied Agent. The International Journal of Human-Computer Studies 59(1-2) (2003)

18. Gratch, J., Marsella, S.: The Architectural Role of Emotion in Cognitive Systems. In: Gray, W. (ed.) Integrated Models of Cognitive Systems. Oxford University Press, Oxford (2008)
19. Adam, C.: Emotions: from psychological theories to logical formalization and implementation in a BDI agent. PhD Thesis (2007)
20. Forlizzi, J., DiSalvo, C., Gemperle, F.: Assistive robotics and an ecology of elders living independently in their homes. Hum.-Comput. Interact. 19, 1 (2004)
21. Dang, T., Letellier-Zarshenas, S., Duhaut, D.: Comparison of recent architectures of emotions. In: 10th International Conference on Control, Automation, Robotics and Vision, ICARCV 2008, Vietnam (2008)
22. Dang, T., Duhaut, D.: Experimentation with GRACE, the Generic Model of Emotions For Computational Applications. In: 2nd Mediterranean Conference on Intelligent Systems and Automation, Tunisia (Best paper award) (2009)

Discrete Dual–Heuristic Programming in 3DOF Manipulator Control

Piotr Gierlak, Marcin Szuster, and Wiesław Żylski

Rzeszów University of Technology
Department of Applied Mechanics and Robotics
8 Powstaców Warszawy St., 35-959 Rzeszów, Poland
{pgierlak,mszuster,wzylski}@prz.edu.pl

Abstract. In this paper we propose a discrete tracking control system for 3 degrees of freedom (DOF) robotic manipulator control. The control system is composed of Adaptive Critic Design (ACD), a PD controller and a supervisory term derived from the Lyapunov stability theory. ACD in Dual–Heuristic Programming (DHP) configuration consists of two structures realized in a form of neural networks (NN): actor - generates a control signal and critic approximates a derivative of the cost function with respect to the state. The control system works on-line, does not require a preliminary learning and uses the 3DOF manipulator dynamics model for a state prediction in ACD structure. Verification of the proposed control algorithm was realized on a SCORBOT 4PC manipulator.

Keywords: Approximate Dynamic Programming, Dual Heuristic Programming, Neural Networks, Robotic Manipulators, Tracking Control.

1 Introduction

The problem of rigid manipulator tracking control is complex, because of nonlinearities in dynamics description of the robot, its potentially not know and changeable parameters and disturbances, that may occur during the movement. That cause a necessity of applying effective computational control algorithms using artificial intelligence methods as for example reinforcement learning algorithms [4], [5], [6], [7], [8], using NNs [3].

In the presented article the NN controller using discrete model–based ACD in DHP configuration [7], [8], is applied to the tracking control of the rigid manipulator. The discrete tracking control system additionally contains PD controller and the supervisory term [9], derived from the Lyapunov stability theorem, that guarantees robustness in a face of disturbances and stability in a NNs learning phase. ACD consists of two structures realized in a form of NNs: actor (ASE - Associate Search Element) generates the optimal control law, and critic (ACE - Adaptive Critic Element) approximates the derivative of the value function with respect to the state. The presented discrete tracking control algorithm does not

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 256–263, 2010.
© Springer-Verlag Berlin Heidelberg 2010

require the preliminary learning, it works on-line and uses the 3DOF manipulator model for the state prediction in DHP structure. Verification of the proposed algorithm was realized on the SCORBOT 4PC manipulator.

The paper is organized as follows: section 2 includes a discrete model of the 3DOF manipulator dynamics. Section 3 presents the discrete model–based ACD in DHP configuration with short description of RVFL NN. Section 4 contains a stability analysis, in section 5 there are presented results of the verification experiment realized on the 3DOF manipulator SCORBOT 4PC. Section 6 summarizes the article.

2 Dynamics Model of 3DOF Manipulator

A scheme of the 3DOF manipulator is shown in Fig. 1.

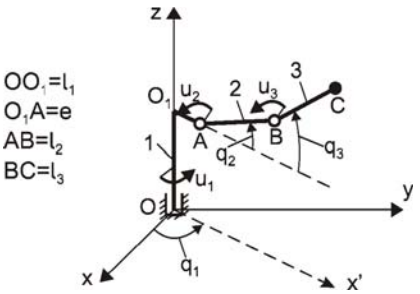


Fig. 1. 3DOF manipulator scheme

The dynamics of the 3DOF manipulator can be written in the Lagrange form

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + F(\dot{q}) + G(q) + \tau_d = u \ , \tag{1}$$

where $q \in R^3$ is the generalized coordinates vector, $M(q)$ is the inertia matrix, $C(q,\dot{q})$ is the Coriolis/centrifugal matrix, $F(\dot{q})$ is the friction vector and $G(q)$ is the gravity vector, τ_d is the vector of bounded disturbances including unstructured and unmodeled dynamics, and u is the vector of control signals. The 3DOF manipulator dynamics model has standard properties [2]. Using the Euler forward difference relation and a state vector in a form $z = \left[z_1^T, z_2^T\right]^T = [q, \dot{q}]^T$, we obtain a discrete notation of the manipulator dynamics in a form

$$\begin{aligned} z_{1_{k+1}} &= z_{1_k} + z_{2_k} h \ , \\ z_{2_{k+1}} &= -M^{-1}\left(z_{1_k}\right)\left[C\left(z_{1_k}, z_{2_k}\right) z_{2_k} + F\left(z_{2_k}\right) + G\left(z_{1_k}\right) + \tau_d - u_k\right] h + z_{2_k} \ , \end{aligned} \tag{2}$$

where k is an index of iteration steps and h is a time discretization parameter.

Let us define the tracking control problem as generating the control law u_k that minimizes a tracking error e_k defined as

$$\begin{aligned} e_{1_k} &= z_{1_k} - z_{1d_k} \ , \\ e_{2_k} &= z_{2_k} - z_{2d_k} \ , \end{aligned} \tag{3}$$

for a desired trajectory z_d in the manipulator join space ($z_k \rightarrow z_{d_k}, k \rightarrow \infty$), and the control system is stable. We assumed a filtered tracking error s_k as

$$s_k = e_{2_k} + \Lambda e_{1_k} \ , \tag{4}$$

where $\Lambda = \Lambda^T > 0$ is a design constant diagonal matrix. Taking into account (3) and (4) we can define a value of the filtered tracking error s_{k+1}

$$s_{k+1} = -Y_f\left(z_{1_k}, z_{2_k}\right) + Y_d\left(z_k, z_{d_{k+1}}\right) - Y_\tau\left(z_{1_k}\right) + M^{-1}\left(z_{1_k}\right) h u_k \ , \tag{5}$$

where

$$\begin{array}{l} Y_f\left(z_{1_k}, z_{2_k}\right) = M^{-1}\left(z_{1_k}\right) h \ \left[C(z_{1_k}, z_{2_k}) z_{2_k} + F(z_{2_k}) + G(z_{1_k})\right] \ , \\ Y_\tau\left(z_{1_k}\right) = M^{-1}\left(z_{1_k}\right) h \, \tau_d \ , \\ Y_d\left(z_k, z_{d_{k+1}}\right) = z_{2_k} - z_{2d_{k+1}} + \Lambda\left[z_{1_k} + z_{2_k} h - z_{1d_{k+1}}\right] \\ = z_{2_k} - z_{2d_k} - z_{3d_k} h + \Lambda\left[z_{1_k} - z_{1d_k} + z_{2_k} h - z_{2d_k} h\right] = s_k + Y_e\left(z_k, z_{d_k}\right) \ , \\ Y_e\left(z_k, z_{d_k}\right) = \Lambda\left[z_{2_k} - z_{2d_k}\right] h - z_{3d_k} h \ , \end{array} \tag{6}$$

and z_{3d_k} is a vector of discrete desired accelerations, that results from the Euler forward difference relation of $z_{2d_{k+1}}$.

The vector $Y_f\left(z_{1_k}, z_{2_k}\right)$ contains all nonlinearities of the 3DOF manipulator.

3 Dual–Heuristic Programming in Control

ACDs [6], [7], [8] derive from the Bellman Dynamic Programming (DP) [1], but thanks to use of adaptive structures like NNs and iterative calculations, they allow to approximate the optimal control law in on-line processes, from the first to the last step of the discrete process (Forward DP).

The objective of the DP algorithm is to determine the optimal control law that minimizes the assumed value function [1], [6], [7], [8], that in general case may have a form

$$V\left(x_k, u_k\right) = \sum_{k=0}^{n} \gamma^k L_C\left(x_k, u_k\right) \ , \tag{7}$$

where x_k is a state, u_k a control signal, n is the last step of finite discrete process, γ is a discount factor ($0 < \gamma < 1$) and $L_C\left(x_k, u_k\right)$ is a local cost in step k.

We assumed the local cost $L_C\left(s_k\right)$ in a form

$$L_C\left(s_k\right) = \frac{1}{2}\left\{s_k^T Q s_k\right\} \ , \tag{8}$$

where Q is a positive defined, fixed diagonal matrix. The objective of ACE-ASE in the tracking control task for the local cost (8), is to generate the control signal that minimizes the filtered tracking error s_k.

In the presented control system we have used DHP algorithm, schematically shown in Fig. 2.a), that consists of the predictive model and two structures realized in the form of NNs:

– the predictive model predicts state $\hat{s}_{k+1}$ in $k+1$ step of a discrete process

$$\hat{s}_{k+1} = s_k - Y_f\left(z_{1_k}, z_{2_k}\right) + Y_e\left(z_k, z_{d_k}\right) + M^{-1}\left(z_{1_k}\right) hu_k \ . \tag{9}$$

– ACE (critic) estimates the derivative of the value function with respect to the state. Critic is realized in a form of three RVFL NNs [3], composed of j neurons with sigmoidal activation functions each, weights of output-layer set equal to zero in an initialization process and randomly chosen fixed weights of first-layer. Scheme of the RVFL NN for ACE$_1$ is shown in Fig. 2.b). ACE generates signal

$$\lambda\left(x_{C_k}, W_{C_k}\right) = W_{C_k}^T S\left(D_C^T x_{C_k}\right) \ , \tag{10}$$

where W_{C_k} is a vector of output-layer weights, D_C is a matrix of fixed input-layer weights, $S(.)$ is a vector of neurons activation functions and $x_{C_k} = [1, s_k]^T$ is an input vector to the ACE NN. ACE weights are adapted by the back propagation method of a difference of the Temporal Difference error in a form [8]

$$\begin{aligned} e_{C_k} = {} & \frac{\partial L_C\left(s_k\right)}{\partial s_k} + \left(\frac{\partial u_k}{\partial s_k}\right)^T \frac{\partial L_C\left(s_k\right)}{\partial u_k} \\ & + \gamma\left[\frac{\partial \hat{s}_{k+1}}{\partial s_k} + \frac{\partial u_k}{\partial s_k}\frac{\partial \hat{s}_{k+1}}{\partial u_k}\right]^T \lambda\left(x_{C_{k+1}}, W_{C_k}\right) - \lambda\left(x_{C_k}, W_{C_k}\right) \ , \end{aligned} \tag{11}$$

where $x_{C_{k+1}}$ includes $\hat{s}_{k+1}$ derived from the predictive model (9).

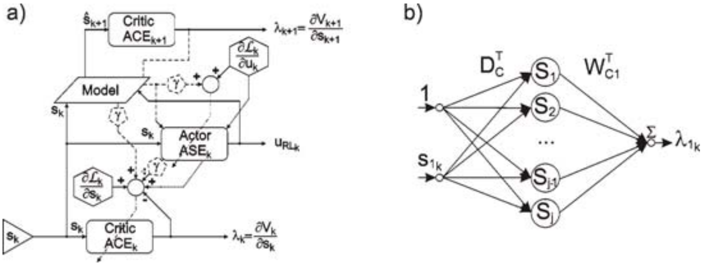


Fig. 2. a) DHP scheme, b) ACE$_1$ RVFL NN scheme

– ASE (actor) approximates the optimal control law and is realized in a form of RVFL NNs similar to ACE NNs, with input vector $x_{S_k} = [1, s_k, z_{2_k}, z_{2d_k}, z_{3d_k}]^T$

$$u_{RL_k} = W_{S_k}^T S\left(D_S^T x_{S_k}\right) \ , \tag{12}$$

where W_{S_k} is a vector of output-layer weights, D_S is a matrix of fixed input-layer weights. ASE minimizes the quality rating in a form [8]

$$e_{S_k} = \frac{\partial L_C\left(s_k\right)}{\partial u_k} + \gamma\left(\frac{\partial \hat{s}_{k+1}}{\partial u_k}\right)^T \lambda\left(x_{C_{k+1}}, W_{C_k}\right) \ . \tag{13}$$

DHP structure consists of ACE in discrete time steps k and $k+1$, ASE and the predictive model (9).

4 Stability Analysis

The presented tracking control system generates the control signal u that consists of the ACE-ASE structure control signal u_{RL}, the supervisory element control signal u_S, the PD control signal u_{PD} and the Y_e control signal. The supervisory term [9], derived from the Lyapunov stability theorem, ensures boundary of the filtered tracking errors s_k and guarantees robustness in a face of disturbances. Schematic structure of the neural controller is shown in Fig. 3.

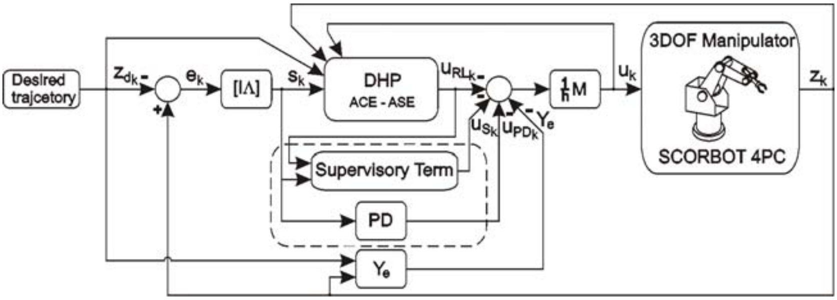


Fig. 3. Schematic structure of the neural controller

We assumed the control signal in a form

$$u_k = \frac{1}{h} M\left(z_{1_k}\right)\left\{-u_{RL_k} + I^* u_{S_k} - K_D\, s_k - Y_e\left(z_k, z_{d_k}\right)\right\} \ , \tag{14}$$

where $u_{PD_k} = K_D s_k$, $K_D = K_D^T > 0$ is a constant gain matrix, I^* is a diagonal matrix, $I^* = 1$ for $|s_{i_k}| \geq \phi_i$ and $I^* = 0$ when $|s_{i_k}| < \phi_i$, ϕ_i is constant, $i = 1, 2, 3$. For equation (14) inserted into (5) and $I^* = 1$ we obtain a description of the closed-loop system in a form

$$s_{k+1} = s_k - Y_f\left(z_{1_k}, z_{2_k}\right) - Y_\tau\left(z_{1_k}\right) - u_{RL_k} - K_D s_k + I^* u_{S_k} \ . \tag{15}$$

We assumed the positive definite Lyapunov candidate function

$$L = s_k^T\, s_k \ . \tag{16}$$

The difference of (16)

$$\varDelta L = s_{k+1}^T\, s_{k+1} - s_k^T\, s_k \ , \tag{17}$$

can be converted to the form

$$\varDelta L = s_k^T\left[s_{k+1} - s_k\right] \ . \tag{18}$$

If we insert (15) into (18) we obtain

$$\varDelta L = s_k^T\left[-K_D\, s_k - Y_f\left(z_{1_k}, z_{2_k}\right) - Y_\tau\left(z_{1_k}\right) - u_{RL_k} + u_{S_k}\right] \ , \tag{19}$$

and can write

$$\Delta L \leq -s_k^T K_D \, s_k + \sum_{i=1}^{2} |s_{i_k}| \left[|Y_{f_i}(z_{1_k}, z_{2_k})| + \left| u_{RL_{i_k}} \right| + b_{d_i} \right] + \sum_{i=1}^{2} s_{i_k} u_{S_{i_k}} \ , \tag{20}$$

where the disturbances are bounded by $|Y_{\tau i}(z_{1_k})| < b_{d_i}$, $b_{d_i} > 0$. If we assume the supervisory term control signal in a form

$$u_{S_{i_k}} = -sgn s_{i_k} \left[F_i + \left| u_{RL_{i_k}} \right| + b_{d_i} \right] \ , \tag{21}$$

we obtain

$$\Delta L \leq 0 \ . \tag{22}$$

where $|Y_{f_i}(z_{1_k}, z_{2_k})| \leq F_i$, and $F_i > 0$, the difference of the Lyapunov function is negative definite. The control algorithm guarantees reduction of $|s_{i_k}|$ for $|s_{i_k}| \geq \phi_i$. For initial condition $|s_{i_{k=0}}| < \phi_i$ we get $|s_{i_k}| < \phi_i$ for $\forall k \geq 0$.

5 Experimental Results

In order to confirm assumed behaviour of the proposed control system, an experiment on the robotic manipulator SCORBOT 4PC was performed. The work station consists of the SCORBOT 4PC manipulator, PC computer with Matlab and DS ControlDesk software, and DS1104 digital signal processing board.

Verification of the proposed discrete tracking control system with model-based DHP structure was realized by the experiment on the SCORBOT 4PC manipulator for a movement of the manipulator's effector (point C) on a desired path shown in Fig. 4.a). Fig. 4.b) shows desired joint angles (q_{1d}, q_{2d}, q_{3d}), and Fig. 4.c) desired joint angular velocities $(\dot{q}_{1d}, \dot{q}_{2d}, \dot{q}_{3d})$.

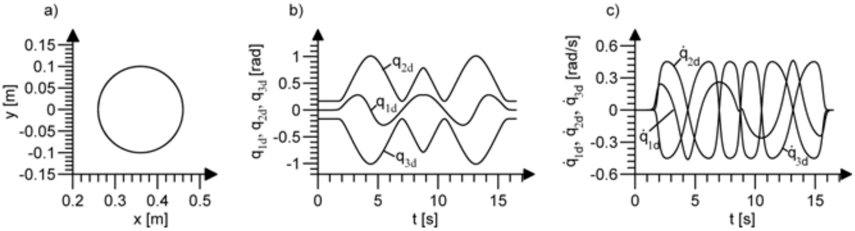


Fig. 4. a) Desired path of the manipulator's effector, b) desired joint angles (q_{1d}, q_{2d}, q_{3d}) [rad], c) desired joint angular velocities $(\dot{q}_{1d}, \dot{q}_{2d}, \dot{q}_{3d})$ [rad/s]

According to the assumed control law (14), the overall control signal u of the neural controller, shown in Fig. 5.a), consists of the DHP ASE control signal u_{RL} (Fig. 5.b)), the PD control signal u_{PD} (Fig. 5.c)), the supervisory term control signal u_S (Fig. 5.d)), and the Y_e control signal. At the beginning of

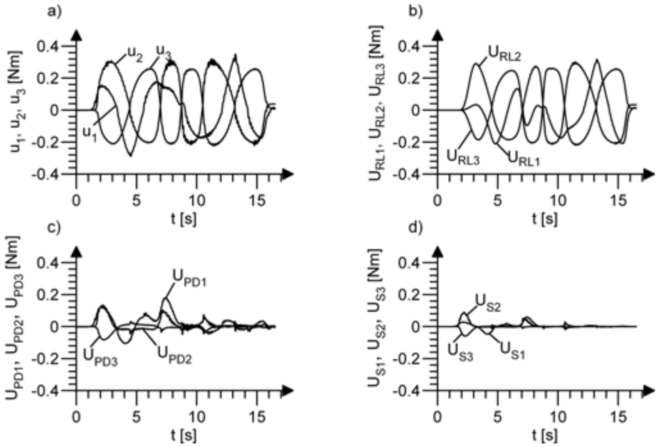


Fig. 5. a) The overall control signals: u_1, u_2 and u_3, b) the ASE NN control signal $U_{RL} = -\frac{1}{h}M^{-1}u_{RL}$, c) the PD control signal $U_{PD} = -\frac{1}{h}M^{-1}u_{PD}$, d) the supervisory term control signal $U_S = -\frac{1}{h}M^{-1}u_S$

the experiment the ACD control signal u_{RL} is not accurate because of NNs' initial weights set to zero, and important part in u have u_{PD} and u_S, but theirs influence decreases during the experiment because of NNs weights adaptation.

The tracking errors of the joint angles (e_1, e_2, e_3) are shown in Fig. 6.a), the tracking errors of the joint angular velocities $(\dot{e}_1, \dot{e}_2, \dot{e}_3)$ are shown in Fig. 6.b). The highest values of the tracking errors appear at the beginning of the experiment and are reduced during ACD NNs weights adaptation. In section 4 there is shown the asymptotical stability of the control system, but in the mechanical systems attainable is the practical stability, what means that the tracking errors converge to the bounded set.

Weights of output-layer of ACD RVFL NNs were set to zero in the initialization process, and adapted during the experiment. In Fig. 6.c) and Fig. 6.d), there are shown values of weights for ASE$_3$ and ACE$_3$ NN. Each of ACD RVFL NNs contains 10 neurons in hidden layer. Farther increase of neurons number did not improved the tracking control quality. Values of NNs weights are bounded and converge to constant values during the adaptation process.

For numerical rating of the realized experiments we used Root Mean Square Error (RMSE) of the filtered tracking errors s_1, s_2 and s_3, defined as: $\varepsilon_{s_j} = \sqrt{\frac{1}{n}\sum_{k=1}^{n} s_{jk}^2}$, j= 1, 2, 3. We have compared quality of the tracking control for the proposed control systems with DHP structure ($\varepsilon_{s_1} = 0.04$, $\varepsilon_{s_2} = 0.032$, $\varepsilon_{s_3} = 0.033$) and only PD controller ($\varepsilon_{s_1} = 0.074$, $\varepsilon_{s_2} = 0.14$, $\varepsilon_{s_3} = 0.194$).We have noticed higher quality of the tracking control for the control system with ACD in a comparison with only PD controller.

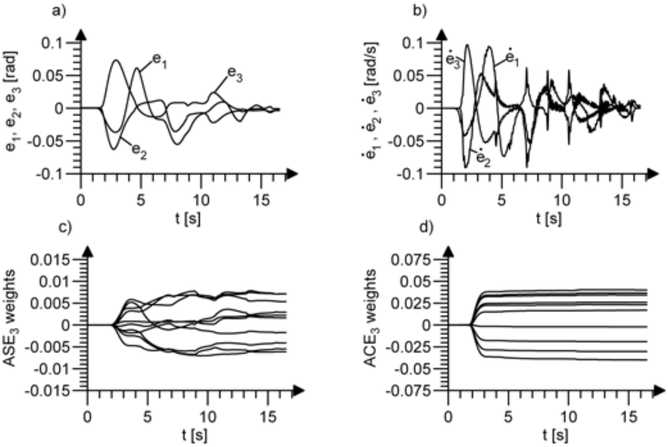


Fig. 6. a) Tracking errors e_1, e_2 and e_3 [rad], b) tracking errors $\dot{e}_1$, $\dot{e}_2$ and $\dot{e}_3$ [rad/s], c) weights of RVFL ASE_3 NN, d) weights of RVFL ACE_3 NN

6 Summary

In the presented article we proposed the control system with model–based ACD in DHP configuration for the tracking control of the robotic manipulator. The control system consists of ACE–ASE structure realized in a form of RVFL NNs, the supervisory term and PD controller. Verification of the proposed control algorithm was realized on SCORBOT 4PC manipulator. The researches confirmed high quality of the tracking control for controller with ACD structur. The values of errors and NNs weights are bounded, the presented control system is stable.

References

1. Bellman, R.: Dynamic Programming. Princeton University Press, New York (1957)
2. de Wit, C.C., Siciliano, B., Bastin, G.: Theory of Robot Control. Springer, London (1996)
3. Giergiel, J., Hendzel, Z., Zylski, W.: Modeling and Control of Wheeled Mobile Robots. WNT, Warsaw (2002) (in Polish)
4. Hendzel, Z.: An Adaptive Critic Neural Network for Motion Control of a Wheeled Mobile Robot. Nonlinear Dynamics 50, 849–855 (2007)
5. Hendzel, Z., Szuster, M.: Discrete Model-Based Dual Heuristic Programming in Wheeled Mobile Robot Control. In: 10th International Conference on Dynamical Systems - Theory and Applications, Lodz, pp. 745–752 (2009)
6. Powell, W.B.: Approximate Dynamic Programming: Solving the Curses of Dimensionality. Willey-Interscience, Princeton (2007)
7. Prokhorov, D., Wunch, D.: Adaptive Critic Designs. IEEE Transactions on Neural Networks 8, 997–1007 (1997)
8. Si, J., Barto, A.G., Powell, W.B., Wunsch, D.: Handbook of Learning and Approximate Dynamic Programming. IEEE Press, Wiley-Interscience (2004)
9. Wang, L.: A Course in Fuzzy Systems and Control. Prentice-Hall, New York (1997)

Discrete Model-Based Adaptive Critic Designs in Wheeled Mobile Robot Control

Zenon Hendzel and Marcin Szuster

Rzeszów University of Technology
Department of Applied Mechanics and Robotics
8 Powstaców Warszawy St., 35-959 Rzeszów, Poland
{zenhen,mszuster}@prz.edu.pl

Abstract. In this paper a discrete tracking control algorithm for a non-holonomic two–wheeled mobile robot (WMR) is presented. The basis of the control algorithm is an Adaptive Critic Design (ACD) in two model-based configurations: Heuristic Dynamic Programming (HDP) and Dual Heuristic Programming (DHP). In proposed control algorithm Actor–Critic structure, composed of two neural networks (NN), is supplied by a PD controller and a supervisory term derived from the Lyapunov stability theorem. The control algorithm works on-line and does not require preliminary learning. Verification of the proposed control algorithm was realized on a WMR Pioneer–2DX.

Keywords: Approximate Dynamic Programming, Dual Heuristic Programming, Heuristic Dynamic Programming, Neural Networks, Tracking Control, Wheeled Mobile Robots.

1 Introduction

Non-holonomic mobile robots are control objects with nonlinear dynamics, which control requires complex computational algorithms. Artificial intelligence (AI) methods as NNs [2], fuzzy logic or reinforcement learning structures [4], [5], [6], [8], [9], [10], [11] are effective computational algorithms that have found practical application in WMR control.

In the literature there are many theoretical results concerning the application of the ACDs in control. This article presents some practical application, where a discrete control algorithm with ACD in HDP and DHP configuration [8], [9], [10], provides tracking control for the WMR in complex transportation task. ACD is composed of two structures: actor (ASE - Associate Search Element) approximates the optimal control law and implements current control policy, critic (ACE - Adaptive Critic Element) evaluates the performance of the current policy, and passes feedback to the actor, which accordingly changes its control policy. ACE approximates the value function in the Bellman Equation [1], [10] (HDP) or it's derivative with respect to the state (DHP). ACE and ASE structures are realized in a form of Random Vector Functional Link (RVFL) [2] NNs (or simple Functional Link NN (FLNN) [7]). Stability of the control system

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 264–271, 2010.
© Springer-Verlag Berlin Heidelberg 2010

is achieved by an additional supervisory control element derived from the Lyapunov stability theory [12], which guarantees robustness in a face of disturbances and stability in ACD NNs learning phase. Presented control algorithm does not require preliminary learning, works on-line and uses the WMR dynamics model for a state prediction in ACD structure. Verification of the proposed control algorithm was realized on the WMR Pioneer-2DX.

The results of researches presented in the article continue authors earlier works related to the WMR tracking control algorithms using NNs [2], fuzzy logic and reinforcement learning methods [4], [5], [6]. The paper is organized as follows: section 1 includes a short introduction into the WMR control problems, section 2 presents a discrete model of the WMR dynamics. In section 3 there are presented discrete model–based ACDs in HDP and DHP configurations with short description of used NNs. Section 4 contains stability analysis, in section 5 results of verification experiments realized on the WMR Pioneer–2DX are presented. Section 6 concludes the research project.

2 The Two–Wheeled Mobile Robot Dynamics

The movement of the non–holonomic WMR with third free rolling castor wheel is analyzed in the xy plane [2]. The WMR is schematically shown in Fig. 1.

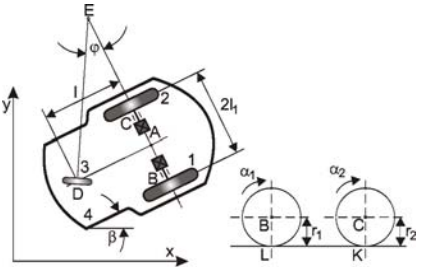


Fig. 1. Schematic diagram of the WMR [2]

We assumed mathematical model of the WMR dynamics using Maggie's mathematical formalism [2], [3]. It involves dynamic equations of the WMR and dynamical properties of executive systems composed of DC motors, reduction gears and position encoders. The dynamics of the WMR can be written as

$$M\ddot{\alpha} + C(\dot{\alpha})\dot{\alpha} + F(\dot{\alpha}) + \tau_d = u \ , \tag{1}$$

where $\alpha = [\alpha_1, \alpha_2]^T$ is a generalized coordinates vector, α_i is an angle of a self-turn for adequate driver wheel, matrices M, $C(\dot{\alpha})$ and vector $F(\dot{\alpha})$ result from Maggie's equations and dynamical properties of the executive systems, τ_d denotes the vector of bounded disturbances, and $u = [u_1, u_2]^T$ is a vector of control signals (motors voltage). Using Euler derivative approximation and a state vector in a form $z = \left[z_1^T, z_2^T\right]^T = \left[\alpha_1, \alpha_2, \dot{\alpha}_1, \dot{\alpha}_2\right]^T$, we obtain a discrete notation of the WMR dynamics in a form

$$\begin{aligned} z_{1_{k+1}} &= z_{1_k} + z_{2_k} h \ , \\ z_{2_{k+1}} &= -M^{-1}\left[C\left(z_{2_k}\right) z_{2_k} + F\left(z_{2_k}\right) + \tau_d - u_k\right] h + z_{2_k} \ , \end{aligned} \tag{2}$$

where h is a time discretization parameter and k is an index of iteration steps. The tracking control problem is defined as searching for the control law, that minimizes tracking errors e_k in the form

$$\begin{aligned} e_{1_k} &= z_{1_k} - z_{d1_k} \ , \\ e_{2_k} &= z_{2_k} - z_{d2_k} \ , \end{aligned} \tag{3}$$

for desired trajectory z_d $\left(z_k \rightarrow z_{d_k}, k \rightarrow \infty\right)$, and the control system remains stable. A filtered tracking error s_k takes the form

$$s_k = e_{2_k} + \Lambda e_{1_k} \ , \tag{4}$$

where Λ is a fixed, positive defined diagonal matrix.
Taking into account (3) and (4) we can define the filtered tracking error s_{k+1}

$$s_{k+1} = -Y_f\left(z_{2_k}\right) + Y_d\left(z_k, z_{d_{k+1}}\right) - Y_\tau + M^{-1} h u_k \ , \tag{5}$$

where

$$\begin{aligned} & Y_f\left(z_{2_k}\right) = M^{-1} h\ \left[C(z_{2_k}) z_{2_k} + F(z_{2_k})\right]\ , Y_\tau = M^{-1} h\, \tau_d \ , \\ & Y_d\left(z_k, z_{d_{k+1}}\right) = z_{2_k} - z_{d2_{k+1}} + \Lambda\left[z_{1_k} + z_{2_k} h - z_{d1_{k+1}}\right] \\ & = z_{2_k} - z_{d2_k} - z_{d3_k} h + \Lambda\left[z_{1_k} - z_{d1_k} + z_{2_k} h - z_{d2_k} h\right] = s_k + Y_e\left(z_k, z_{d_k}\right)\ , \\ & Y_e\left(z_k, z_{d_k}\right) = \Lambda\left[z_{2_k} - z_{d2_k}\right] h - z_{d3_k} h \ , \end{aligned} \tag{6}$$

and z_{d3_k} is a vector of desired accelerations, that results from forward difference relation of $z_{2d_{k+1}}$. The vector $Y_f\left(z_{2_k}\right)$ contains all nonlinearities of the WMR.

3 Adaptive Critic Designs in Control

ACDs are a group of Forward Dynamic Programming (FDP) methods [8], [9], [10], which derives from the Bellman DP [1]. The objective of the DP is to determine the optimal control law that minimizes the value function [1], [8], [9], [10], which is the function of a state x_k and a control u_k in general case

$$V\left(x_k, u_k\right) = \sum_{k=0}^{n} \gamma^k L_C\left(x_k, u_k\right) \ , \tag{7}$$

where n is the last step of finite discrete process, γ is a discount factor $(0 < \gamma < 1)$ and $L_C\left(x_k, u_k\right)$ is a local cost in step k.

We assumed the local cost $L_C\left(s_k\right)$ in the form

$$L_C\left(s_k\right) = \frac{1}{2}\left\{s_k^T Q s_k\right\} \ , \tag{8}$$

where Q is a fixed, positive defined diagonal matrix. The objective of ACDs in the tracking control task for the local cost (8), is to generate the control law that minimizes the filtered tracking error s_k.

We have used two model–based ACDs to solve the tracking control problem.

3.1 Heuristic Dynamic Programming with FLNN ACE NN (HDP$_F$)

HDP$_F$ algorithm consists of a predictive model and two structures:
– the model predicts state $\hat{s}_{k+1}$ in $k+1$ step of a discrete process

$$\hat{s}_{k+1} = s_k - Y_f\left(z_{2_k}\right) + Y_e\left(z_k, z_{d_k}\right) + M^{-1} h u_k \ . \tag{9}$$

– critic (ACE) estimates the optimal value function $V\left(s_k\right)$ and is realized in a form of two FLNN NNs [7], with two functional links each. ACE generates a signal in the form

$$\hat{V}_F\left(s_k, W_{FC_k}\right) = W_{FC_k}^T S_F\left(s_k\right) \ , \tag{10}$$

where W_{FC_k} is a vector of output-layer weights, $S_F\left(s_k\right) = \left[s_k, s_k^2\right]^T$ is a vector of neurons activation functions. The input vector to the ACE FLNN contains s_k because the value function $V\left(s_k\right)$ depends only on the filtered tracking error. Critics weights are adapted by the back propagation method of the Temporal Difference error [9]

$$e_{TD_k} = L_C\left(s_k\right) + \hat{V}_F\left(\hat{s}_{k+1}, W_{FC_k}\right) - \hat{V}_F\left(s_k, W_{FC_k}\right) \ , \tag{11}$$

where $\hat{s}_{k+1}$ is derived from the predictive model (9).
– actor (ASE) estimates the optimal control law and is realized in a form of two RVFL NNs

$$u_{RL_k} = W_{S_k}^T S\left(D_S^T x_{S_k}\right) \ , \tag{12}$$

where W_{S_k} is a vector of output-layer weights, D_S is a matrix of fixed input-layer weights, $S(.)$ is a vector of sigmoidal neurons activation functions and $x_{S_k} = \left[1, s_k, z_{2_k}, z_{2d_k}, z_{3d_k}\right]^T$.

ASE estimates the optimal control law by the back propagation of a quality rating in a form [9]

$$e_{S_k} = \frac{\partial L_C\left(s_k\right)}{\partial u_k} + \gamma \left(\frac{\partial \hat{s}_{k+1}}{\partial u_k}\right)^T \frac{\partial \hat{V}_F\left(\hat{s}_{k+1}, W_{FC_k}\right)}{\partial \hat{s}_{k+1}} \ . \tag{13}$$

3.2 Heuristic Dynamic Programming with RVFL ACE NN (HDP$_R$)

HDP$_R$ algorithm, in detail presented in [6], consists of the predictive model and two structures: ACE and ASE, realized in the form of RVFL NN. Both neural networks are adapted by the back propagatin method of quality ratings (11) and (13) for ACE and ASE respectively.

3.3 Dual Heuristic Programming (DHP)

DHP algorithm, in detail described in [5], consists of the predictive model and two structures: ACE and ASE, realized in the form of RVFL NN. Both neural networks are adapted by the back propagation method. In DHP algorithm, ACE approximates the derivative of the value function with respect to the state.

4 Stable Neural Tracking Control System

In the presented tracking control system the control signal of ACE-ASE structure u_{RL} is supplied by the additional supervisory element [12] derived from the Lyapunov stability theorem u_S, PD control signal u_{PD} and Y_e control signal. Schematic structure of the neural controller is shown in Fig. 2.

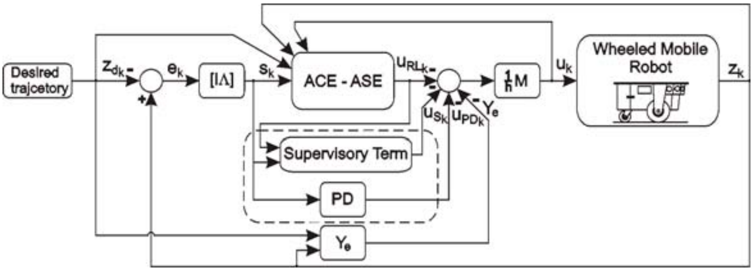


Fig. 2. Schematic structure of the neural controller

The supervisory element ensures stability, which means that the filtered tracking errors s_k are bounded. We assumed the control signal in a form

$$u_k = \frac{1}{h} M \left\{ -u_{RL_k} + I^* u_{S_k} - K_D \, s_k - Y_e \left(z_k, z_{d_k} \right) \right\} \ , \tag{14}$$

where $u_{PD_k} = K_D s_k$, K_D is a fixed, positive defined diagonal matrix, I^* is a diagonal matrix, $I^* = 1$ for $|s_{i_k}| \geq \phi_i$ and $I^* = 0$ when $|s_{i_k}| < \phi_i$, ϕ_i is constant, $i = 1, 2$. For $I^* = 1$, and (14) inserted into (5), we obtain

$$s_{k+1} = s_k - Y_f \left(z_{2_k} \right) - Y_\tau - u_{RL_k} - K_D s_k + u_{S_k} \ . \tag{15}$$

For the positive definite Lyapunov candidate function

$$L = s_k^T \, s_k \ , \tag{16}$$

the difference

$$\Delta L = s_{k+1}^T \, s_{k+1} - s_k^T \, s_k \ , \tag{17}$$

can be converted to the form

$$\Delta L = s_k^T \left[s_{k+1} - s_k \right] \ . \tag{18}$$

Substituting (15) into (18) we obtain

$$\Delta L = s_k^T \left[-K_D \, s_k - Y_f \left(z_{2_k} \right) - Y_\tau - u_{RL_k} + u_{S_k} \right] \ . \tag{19}$$

If we assume, that the disturbances are bounded by $|Y_{\tau_i}| < b_{d_i}$, $b_{d_i} > 0$, we can write

$$\Delta L \leq -s_k^T K_D \, s_k + \sum_{i=1}^{2} |s_{i_k}| \left[|Y_{f_i} \left(z_{2_k} \right)| + \left| u_{RL_{i_k}} \right| + b_{d_i} \right] + \sum_{i=1}^{2} s_{i_k} u_{S_{i_k}} \ . \tag{20}$$

For the supervisory term control signal in a form

$$u_{S_{ik}} = -sgns_{ik} \left[F_i + \left| u_{RL_{ik}} \right| + b_{d_i} \right] \ , \tag{21}$$

where $|Y_{f_i}(z_{2_k})| \leq F_i$, and $F_i > 0$, we obtain

$$\Delta L \leq 0 \ . \tag{22}$$

The difference of the Lyapunov function is negative definite. The designed control algorithm guarantees reduction of $|s_{i_k}|$ for $|s_{i_k}| \geq \phi_i$. For initial condition $|s_{i_{k=0}}| < \phi_i$ we get $|s_{i_k}| < \phi_i$ for $\forall k \geq 0$, $i = 1, 2$.

5 Experiments Results

Verification of the proposed control algorithm was realized by a series of experiments on the Pioneer–2DX, for movement of a chosen point A of the WMR on a desired path shown in Fig. 3.a). Fig. 3.b) shows desired angles of a self turn α_{d1} and α_{d2} for wheels 1 and 2, Fig. 3.c) desired angular velocities $\dot{\alpha}_{d1}$, $\dot{\alpha}_{d2}$. In our experiments there were two parametric disturbances (marked on diagrams by ellipses), realized as an increase ($\Delta m_{WMR} = 3.8$[kg] in $t_1 = 9$ [s]), and next decrease ($\Delta m_{WMR} = -3.8$[kg] in $t_2 = 30$ [s]) of the load transported by the WMR.

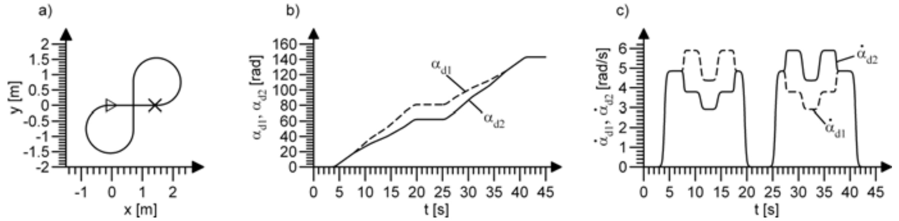


Fig. 3. a) Desired path of the point A movement, b) desired angles of a self-turn α_{d1} and α_{d2} [rad], c) desired angular velocities $\dot{\alpha}_{d1}$ and $\dot{\alpha}_{d2}$ [rad/s]

We have tested four control systems:
– PD controller - control signal consists only of u_{PD},
– HDP$_R$ control system consists of u_{RL} (HDP with RVFL ACE), u_S and u_{PD},
– HDP$_F$ control system consists of u_{RL} (HDP with FLNN ACE), u_S and u_{PD},
– DHP control system consists of u_{RL} (DHP structure), u_S and u_{PD}.
According to the assumed control law (14), the overall control signal u shown in Fig. 4.a), consists of ACE-ASE control signal u_{RL}, supervised control signal u_S, PD controller signal u_{PD} (Fig. 4.b)), and Y_e signal. In Fig. 4.c) there are shown tracking errors e_1 and $\dot{e}_1$ for the first wheel of the WMR. Fig. 4.d) shows desired (x_{Ad_k}, y_{Ad_k}) and realized path of the point A movement (x_{A_k}, y_{A_k}), Fig. 4.e)

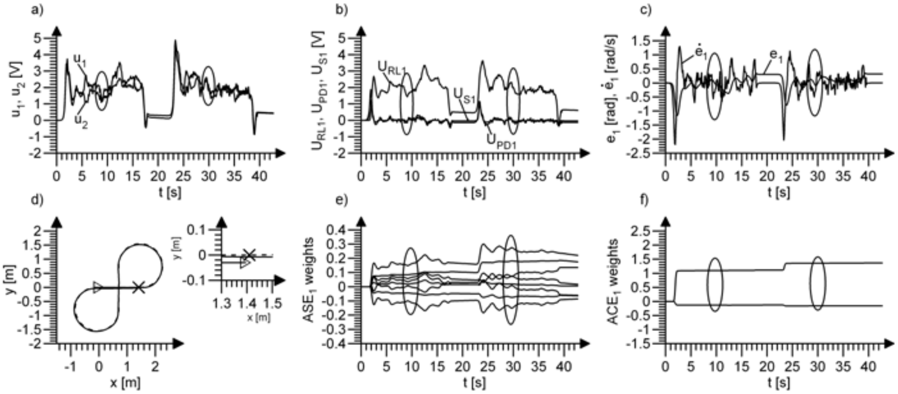


Fig. 4. a) Control signals: u_1 for 1. and u_2 for 2. wheel, b) ASE NN control signal $U_{RL1} = -\frac{1}{h}M^{-1}u_{RL1}$, PD control signal $U_{PD1} = -\frac{1}{h}M^{-1}u_{PD1}$, and the supervisory term control signal $U_{S1} = -\frac{1}{h}M^{-1}u_{S1}$, c) tracking errors e_1 [rad] and $\dot{e}_1$ [rad/s], d) desired (dashed line) and realized path of the point A movement, e) weights of RVFL ASE$_{F1}$ neural network for HDP$_F$ structure and wheel 1, f) weights of FLNN ACE$_{F1}$ neural network for HDP$_F$ structure.

and f) present values of HDP$_F$ ASE RVFL and ACE FLNN weights respectively. Each of ASE RVFL NNs contains 8 neurons and $h = 0.01[s]$.

At the moment of the parametric disturbances we can observe temporary increase of the tracking errors values. The tracking errors are reduced, while the compensation of the WMR nonlinearities becomes more adequate. Weights of NNs are bounded and converge to the fixed values.

For numerical rating of the realized experiments quality ratings were chosen:
– distance between destination and finish of the WMR movement point $d_k = \sqrt{(x_{Ad_k} - x_{A_k})^2 + (y_{Ad_k} - y_{A_k})^2}$ [m], for k=n, n - number of iteration steps,
– Root Mean Square Error (RMSE) of the point A position errors ε_d,
– average RMSE of the filtered tracking errors s_1 and s_2, defined as: $\varepsilon_{s_{avg}} = 0.5\left(\varepsilon_{s1} + \varepsilon_{s2}\right)$, where $\varepsilon_{s_j} = \sqrt{\frac{1}{n}\sum_{k=1}^{n} s_{jk}^2}$, j=1,2, $\varepsilon_d = \sqrt{\frac{1}{n}\sum_{k=1}^{n} {d_k}^2}$ [m].
Values of quality ratings are shown it Tab. 1.

Table 1. Values of the quality ratings

control system	PD	HDP$_R$	HDP$_F$	DHP
PD $(K_D/\Lambda\ )$	$2h/1$	$2h/1$	$2h/1$	$2h/1$
n	429	428	428	429
d_n	0.2263	0.05387	0.03332	0.02209
ε_d	0.2974	0.1335	0.0531	0.0378
$\varepsilon_{s_{avr}}$	4.6304	0.5950	0.4984	0.3560

We have compared quality of the tracking control for the proposed control systems and PD controller. On the basis of the obtained results we can notice higher quality of the tracking control for the control system with ACD in a comparison with only PD controller. Proposed FLNN ACE used in HDP$_F$ structure let us improve tracking control quality of HDP$_F$ in comparison with HDP$_R$.

6 Conclusion

We proposed the discrete tracking control system with ACD structure. The presented control system consists of ACE–ASE algorithm supplied by PD controller and the supervisory term, which guarantee stable realization of tracking. Verifications of the proposed control algorithm was realized on the WMR Pioneer–2DX for complex transportation task with change of the load during the movement. The researches pointed out higher quality of the tracking control for the control system with ACD in a comparison with only PD controller. We have notice worse quality of tracking control for HDP$_R$ structure compared to DHP, when using RVFL NN in ACE structure. Proposed FLNN ACE used in HDP$_F$ structure let us improve tracking control quality of HDP$_F$. Used FLNN is less computationally complex. The proposed discrete tracking control algorithm is stable in a face of disturbances, values of the tracking errors and weights of NNs are bounded.

References

1. Bellman, R.: Dynamic Programming. Princeton University Press, New York (1957)
2. Giergiel, J., Hendzel, Z., Zylski, W.: Modeling and Control of Wheeled Mobile Robots. WNT, Warsaw (2002) (in Polish)
3. Giergiel, J., Zylski, W.: Description of Motion of a Mobile Robot by Maggie's Equations. Journal of Theoretical and Applied Mechanics 43, 511–521 (2005)
4. Hendzel, Z.: An Adaptive Critic Neural Network for Motion Control of a Wheeled Mobile Robot. Nonlinear Dynamics 50, 849–855 (2007)
5. Hendzel, Z., Szuster, M.: Discrete Model-Based Dual Heuristic Programming in Wheeled Mobile Robot Control. In: 10th International Conference on Dynamical Systems - Theory and Applications, Lodz, pp. 745–752 (2009)
6. Hendzel, Z., Szuster, M.: Heuristic Dynamic Programming in Wheeled Mobile Robot Control. In: 14th IFAC International Conference on Methods and Models in Automation and Robotics, Miedzyzdroje, pp. 37–41 (2009)
7. Krishnaiah, D., et al.: Application of Ultrasonic Waves Coupled with Functional Link Neural Network for Estimation of Carrageenan Concentration. International Journal of Physical Sciences 3, 90–96 (2008)
8. Powell, W.B.: Approximate Dynamic Programming: Solving the Curses of Dimensionality. Willey-Interscience, Princeton (2007)
9. Prokhorov, D., Wunch, D.: Adaptive Critic Designs. IEEE Transactions on Neural Networks 8, 997–1007 (1997)
10. Si, J., Barto, A.G., Powell, W.B., Wunsch, D.: Handbook of Learning and Approximate Dynamic Programming. IEEE Press, Wiley-Interscience (2004)
11. Syam, R., Watanabe, K., Izumi, K.: Adaptive Actor-Critic Learning for the Control of Mobile Robots by Applying Predictive Models. Soft Computing 9, 835–845 (2005)
12. Wang, L.: A Course in Fuzzy Systems and Control. Prentice-Hall, New York (1997)

Using Hierarchical Temporal Memory for Vision-Based Hand Shape Recognition under Large Variations in Hand's Rotation

Tomasz Kapuscinski

Rzeszow University of Technology,
Department of Computer and Control Engineering,
W. Pola 2, 35-959 Rzeszow, Poland
tomekkap@prz-rzeszow.pl

Abstract. Hierarchical Temporal Memory (HTM), a new computational paradigm based on cortical theory, has been applied to vision-based hand shape recognition under large variations in hand's rotation. HTM's abilities to build invariant object representations and solve ambiguities have been explored and quite promising results have been achieved for the difficult recognition task. The four-component edge orientation histograms calculated from the Canny edge images, have been proposed as the output of the HTM sensors. The two-layer HTM, with 16x16 nodes in the first layer and 8x8 in the second one, has been experimentally selected as the structure giving the best results. The 8 hand shapes, generated for 360 different rotations, have been recognized with efficiency up to 92%.

Keywords: hand gestures recognition, hierarchical temporal memory, human-computer interaction.

1 Introduction

Hand gestures constitute the important component of the non-verbal communication and therefore should be one of the input modalities in the advanced human-computer interaction [1]. The non-intrusive gesture recognition in a vision system allows for human-like, natural communication with the computer. Gestures can be also used as the supplementary, supportive channel when the voice command recognition is ambiguous due to the noisy environment. Together with the face expressions, they convey an additional information (e.g. human emotions) that can be used to adjust the various interface's parameters and adapt the system behaviour to the user's state. Persons with hearing impairment will also benefit from such interfaces because the sign language is their basic communication tool. Moreover performing some actions is more intuitive using gestures than voice commands or even mouse and keyboard (e.g. rotating a 3D object). The immersive computer games are another prospective application areas. Therefore, there is a need for developing the robust automatic gestures recognition that will work with as little constraints as possible.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 272–279, 2010.
© Springer-Verlag Berlin Heidelberg 2010

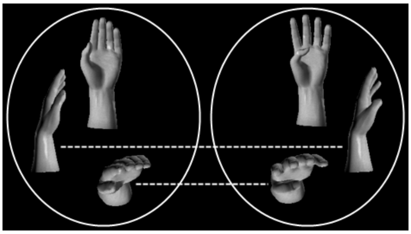


Fig. 1. Images of two different gestures registered at three different hand rotations (dashed lines connect the inter-class images that are very similar)

Humans perform gestures spontaneously, some of them are even done unwittingly, without thinking about precise hand's positioning, maintaining the exact rotation angles and constant distance to the interlocutor. Modern human-computer interaction should allow for the communication in the similar manner. Otherwise it will be non-ergonomic and its usage will be tiring.

Invariant hand gestures recognition in a vision-based system is the great challenge. Interpreting the 2D images of the 3D objects observed at varying distances and rotations leads to many ambiguities. As it is shown in Fig.1, in such a case, images from different categories may have more overlap in the pixel space than images from the same category. Therefore there is a need for tool that will be able to build the invariant object representation and deal well with ambiguities.

For the visual understanding of human gestures, a number of recognition techniques have so far been proposed, e.g. [2,3,4,5,6,7]. A perfect method for hand shape recognition does not yet exist, each solution has specific strengths and weaknesses which may be more or less important depending on the application. This paper deals with large hand rotations, so taking the view independence as the comparison criterion, one can note that most data-driven methods and methods using an appearance model can only represent limited viewing and pose variation in a single model. Therefore these methods require multiple models to represent all possible views. Due to the sensitivity to changes in view, systems using low-level image features require a lot of training examples to model just a single view. Methods using a 3D model for training or fitting are better suited for view independence but they impose the scalability problem - adding the new gesture requires preparing the new 3D model(s).

Several clues can be obtained by analyzing how humans solve the vision problem [8]. We can pretty well recognize objects from a single static snapshot despite changes in location, size, lighting conditions and in the presence of deformations and large amounts of noise. However when we learn we use time extensively. We observe continuously varying data in order to ascertain generalization characteristics. As children, we see objects over and over again, in many different lighting conditions, sizes, and views and finally we learn reliably how to identify them. When we are confronted with a new and confusing object, we pick it up and move it about in front of our eyes. As the object moves, the patterns on our retina changes and our cortical system is able to build its invariant model.

In other words, to recognize the static pictures we have to train on moving images.

The invariant information about the learned object is not stored in one particular location. It is distributed across many nodes up and down the hierarchy in the human neocortex. Low-level visual details are stored in low-level nodes, and high-level structure is stored in higher-level nodes. Objects models created in our brain are composed from the smaller reusable units. The leads to good scalability while adding the new objects but also gives mechanism to solve ambiguities. As the information propagates up the hierarchy, it becomes more stable and unambiguous.

Recently the Hierarchical Temporal Memory (HTM), the new computing paradigm, replicating the aforementioned structural and algorithmic properties of the human neocortex, has been developed [9,10]. HTM has been already applied by the author to the dynamic sign expressions recognition. In the paper [11] the ability to build the object's representation from the smaller parts has been explored in order to achieve subunits based recognition of the dynamic Polish Sign Language gestures. In this paper, a quite different problem of the static gestures recognition under large variations in hand's rotations, and two different capabilities of the HTM: ability to build invariant object's model and resolving ambiguities are addressed. Additional motivation, except this aforementioned in the beginning, is that rotation invariant hand shape recognition is needed to build tool that will convert an unknown Polish Sign Language gesture into so called gestogram transcription [12]. This will allow for creating subunits based sign gestures recognition strongly grounded on the linguistic research.

The paper is organized as follows. Section 2 contains a brief overview of HTM. Section 3 defines the problem and gives details of the proposed approach. Results of recognition is summarized in section 4. Section 5 concludes the paper.

2 Hierarchical Temporal Memory Concept

Hierarchical Temporal Memory is a technology that replicates the structural and algorithmic properties of the neocortex. HTM is organized as a tree-shaped hierarchy of nodes. All objects in the world have a structure. This structure is hierarchical in both space and time. HTM is also hierarchical in both space and time, and therefore it can efficiently represent the structure of the world.

HTM receives the spatio-temporal pattern coming from the senses. Through a learning process it discovers what the causes are and develops internal representations of the causes in the world (Fig. 2). After an HTM has learned what the causes in its world are and how to represent them, it can perform inference. Inference is similar to pattern recognition. Given a novel sensory input stream, the HTM will infer what known causes are likely to be present in the world at that moment.

Each node in HTM implements a common learning and memory function. The basic operation of each node is divided into two steps (see Fig. 3). The first step is to assign the node input pattern to one of a set of quantization

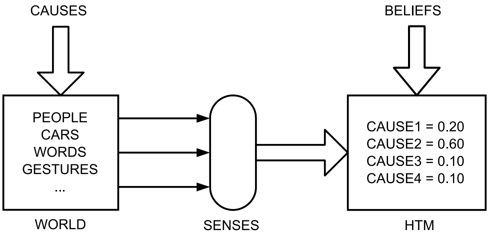


Fig. 2. HTM concept

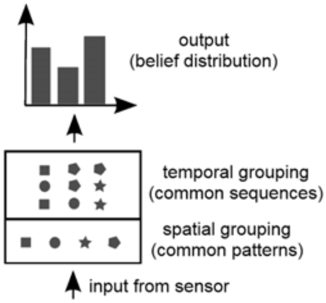


Fig. 3. HTM node operation: learning the invariant representations

points (spatial grouping). The node decides how close (spatially) the current input is to each of its quantization points. In the second step, the node looks for common sequences of these quantization points (temporal grouping). Discovered causes are connected to the common sequences stored in the node's temporal sublayer. These common sequences contains all learned variations of the observed object. Given the static input we can obtain the distribution over all causes (in the noiseless case it is based on the occurrence of the given static pattern in the particular stored sequence). The detailed description can be found in [13].

Described node's operation allows for creating the invariant representations but does not solve the possible ambiguities. The obtained distributions can be very flat. This is where hierarchy comes into the picture. HTM rapidly resolves conflicting or ambiguous input as information flows up the hierarchy. The variation of belief propagation technique is used to do inference. The sensory data imposes a set of beliefs at the lowest level, and by the time the beliefs propagate to the highest level. The highest level nodes show what highest level causes are most consistent with the inputs at the lowest levels. Imagine a network with three nodes, a parent node and two children nodes (Fig. 4). The first child node believes that it is seeing rather B, which if in fact wrong response. The sibling

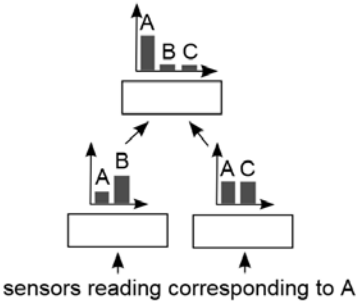


Fig. 4. Beliefs propagation in hierarchy helps to solve ambiguities

node's answer is ambiguous, it is not certain if it sees A or C. Parent node decides with high certainty that A is present. It chose A because this belief is the only one that is consistent with its inputs. It made this choice even though A was not the most likely beliefs of the child nodes. Belief propagation in the hierarchy assures that the system very quickly settle.

3 Problem Definition and Proposed Solution

The 8 static hand gestures have been considered (Fig. 5). Images have been generated using the graphical tool Virtual Hand Studio, which gives pretty realistic views of the hand [14]. Each gesture has been shown at 4 different realizations, 35 hand rotations around its vertical axis (from 0 to 350 degrees with the regular step of 10) and 10 rotations around its horizontal axis (for 0 to 90 with the step of 10). This gives 8 x 4 x 36 x 90 = 103680 images, 12960 for each gesture (Fig. 6). The edge images obtained with the Canny edge detector have been used as the HTM network input (Fig. 7). The difficulty of this problem can be understood by looking at Fig. 1, 6, and 7. The similarities between views selected from the different categories are bigger than intra-class correlations.

After many experiments with the network structure and parameters the following two-layer HTM has been proposed (Fig. 8). The sensors' layer consists of 16 x 16 elements. The edge orientation histogram, calculated from the 20x15 pixels image window covered by the sensor, has been chosen as the sensor's output.

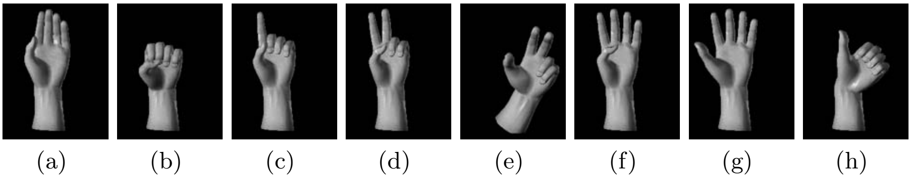


Fig. 5. Hand gestures used in experiments: (a) *flat hand*, (b) *fist*, (c) *1 finger*, (d) *2 fingers*, (e) *3 fingers*, (f) *4 fingers*, (g) *5 fingers*, (h) *thumb*

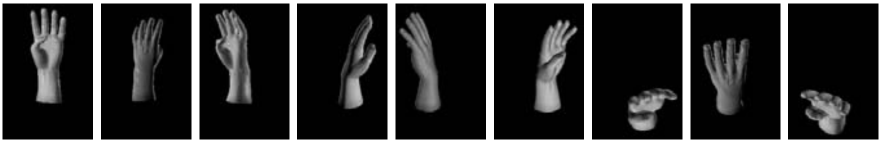


Fig. 6. Selected images for the gesture *4 fingers*

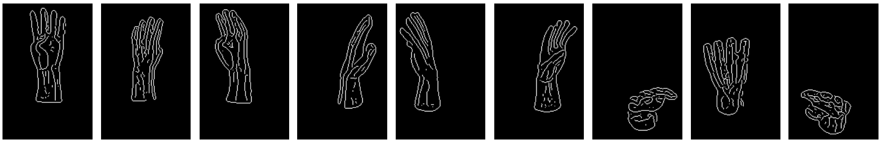


Fig. 7. Selected edge images for the gesture *4 fingers*

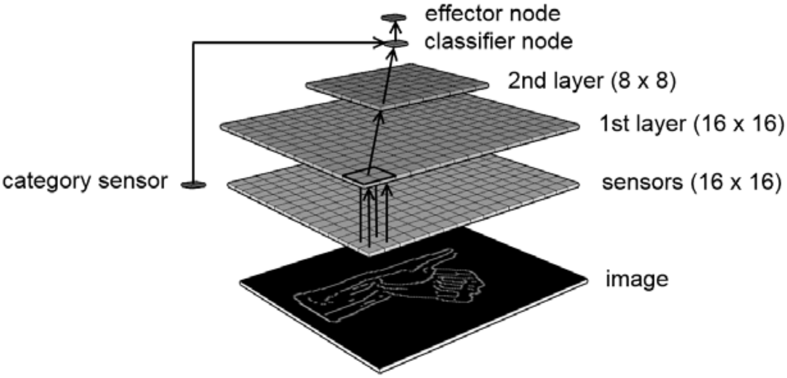


Fig. 8. The HTM topology

Only four different edge orientations are distinguished: horizontal, vertical and two slanting. The category sensor is connected only during the learning stage and it supplies the information about the gesture to which the current sensors reading belongs. The temporal hierarchy is mapped by putting two layers containing 16x16 and 8x8 nodes respectively. Each of these layers consists of two sub-layers making spatial and temporary grouping appropriately [9,10]. Above these layers the NN classifier and effector nodes are put. The effector node writes the results to the file. The spatial hierarchy manifests itself in linking the nodes between the particular layers.

4 Experiments

The Numenta Platform for Intelligent Computing (NuPIC) has been used [15]. NuPIC implements a hierarchical temporal memory system. To make the experimentations easier the auxiliary scripts, dedicated to some specific tasks, as data organization, network structure creating and results analysis, have been developed in Matlab. The whole images set W has been divided into four mutually separated subsets Z_1, Z_2, Z_3, Z_4. Each subset contains images for one realization of each gesture under all possible rotations. To achieve good generalization properties, at the learning stage movies (image sequences) showing the given gesture while rotating have been presented. Sample results of recognition are given in table 1.

Table 1. Hand shape recognition rates [%]: variant a - training on $W - Z_i$, testing on Z_i, variant b - training on Z_i, testing on $W - Z_i$

	variant a	variant b
Z_1	91.63	76.88
Z_2	91.77	76.94
Z_3	92.74	77.09
Z_4	92.26	78.04
mean	92.10	77.24

The obtained results are quite promising and HTM seems to be the relevant tool for vision-based hand shape recognition under large variation in rotation. Please note, that in the variant b the hand shape for given rotation angles has been presented only once during the training stage. Besides, it is worthwhile noticing that for some of the considered images even human has difficulty to correctly recognize the shape (see Fig. 1).

5 Conclusions and Future Work

Hierarchical Temporal Memory has been used for recognition of hand shape under large variations in rotation. This tool has been chosen because it replicates the structural and algorithmic properties of the human neocortex, therefore it should be able to recognize the gestures in a way humans do. During the training HTM builds the invariant object's representations. At the interference phase it deals pretty well with ambiguities. Moreover the hierarchical structure of the HTM allows for sharing the representations. Different objects in the higher level can be composed from the same lower level parts. This should lead to the better scalability and storage efficiency. To author's knowledge this is the first application of HTM to static hand gestures recognition. Obtained results are quite promising. However, further refining the memory structure and parameters is needed. Further works may also include testing the recognition under changes in size, lighting and at presence of noise. Some interesting additional humans capabilities may also be explored, e.g. the mechanism of covering attention. Human system is able to limit the perceptual experience to a variable size area in the center of the visual field. This may be applied to HTM as well. There is no need to convey information from all input sensors but only from those areas that contain the hand's edges.

Acknowledgment

This research was supported by the Polish Ministry of Higher Education under grant N N516 369736.

References

1. Kraiss, K.F.: Advanced Man-Machine Interaction. Springer, Berlin (2006)
2. Athitsos, V., Sclaroff, S.: Estimating 3D Hand Pose from a Cluttered Image. In: Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2003), vol. 2, pp. 432–440 (2003)
3. Guan, H., Chang, J.S., Chen, L., Feris, R.S., Turk, M.: Multi-view appearance-based 3d hand pose estimation. In: Proc. of Conference on Computer Vision and Pattern Recognition Workshop, pp. 154–155. IEEE Computer Society, Washington (2006)
4. Tanimoto, T., Hoshino, K.: Real Time Posture Estimation of Human Hand for Robot Hand Interface. In: Second International Symposium on Universal Communication, pp. 303–308. IEEE Computer Society, Los Alamitos (2008)
5. Lee, S.U., Cohen, I.: 3D Hand Reconstruction from a Monocular View. In: 17th International Conference on Pattern Recognition (ICPR 2004), vol. 3, pp. 310–313 (2004)
6. Ong, E., Bowden, R.: A boosted classifier tree for hand shape detection. In: Proc. of 6th International Conference on Automatic Face and Gesture Recognition, pp. 889–894 (2004)
7. Stenger, B., Thayananthan, A., Torr, P., Cipolla, R.: Hand pose estimation using hierarchical detection. LNCS, pp. 105–116. Springer, Heidelberg (2004)
8. Hawkins, J.: Learn like a human. IEEE Spectrum (2007)
9. Hawkins, J., Blakeslee, S.: On Intelligence. Times Books, New York (2004)
10. Hawkins, J., Dileep, G.: Hierarchical Temporal Memory, Concepts, Theory and Terminology. Numenta Inc. (2006), http://www.numenta.com/Numenta_HTM_Concepts.pdf
11. Kapuscinski, T., Wysocki, M.: Using Hierarchical Temporal Memory for Recognition of Signed Polish Words. In: Kurzynski, M., Wozniak, M. (eds.) Computer Recognition Systems: Advances in Intelligent and Soft Computing, pp. 355–362. Springer, Heidelberg (2009)
12. Szczepankowski, B.: Sign language in school. WSiP, Warsaw (1988) (in Polish)
13. Dileep, G., Bobby, J.: The HTM Learning Algorithms. Numenta Inc. (2007), http://www.numenta.com/for-developers/education/algorithms.php
14. Virtual Hand Studio. 3D Virtual Figure Drawing, Anatomy Softwares and Anatomical models. D-AnatomyStore, CloudStars, http://d-anatomystore.com/
15. Numenta Platform for Intelligent Computing (NuPIC). Numenta Inc., http://www.numenta.com

Parallel Graph Transformations with Double Pushout Grammars

Leszek Kotulski and Adam Sędziwy

AGH University of Sciences and Technology, Institute of Automatics,
al. Mickiewicza 30, 30-059 Kraków, Poland
{kotulski,sedziwy}@agh.edu.pl

Abstract. Multiagent systems implementing artificial intelligence systems, require a formal representation to specify and simulate their properties and behavior. Double pushout graph grammars posses a very high expressive power; the possibility of the use of parallel graph transformations in a distributed environment make them useful in this area thanks to application of the complementary graphs concept. The mentioned idea is formally introduced and the polynomial computational complexity of underlying algorithms is proved.

1 Introduction

Multiagent systems are one of the fundamental tools of an artificial intelligence systems implementation. Each an agent has the capabilities for solving a given problem basing on incomplete (local) information. There is no global system control and a data is decentralized so it is easy to deploy the agents into a distributed environment and evaluate their behavior in a parallel, asynchronous way [12]. On the other hand the lack of the formal representation of a global knowledge of an agent system causes that an attempt to solve problems of specification and simulation of an agent system behavior fails in a general case. One can find only examples of the partial solutions such as specification of a local agents behavior only [10] or the specification of a communication between them [11]. In the paper we suggest to solve this issue by joining two concepts: double pushout (DPO) graph transformations [2] and complementary graphs parallel transformations [6]. DPO approach (reviewed in Section 2) was successfully applied to specify the concurrent semantics [1] or modeling concurrent, mobile and coordinated systems [9]. Currently its attractiveness grows, since there was formally proven its equivalency to triple graph grammars [5], that has been adequate basis to specify visual, formal and bidirectional model transformations between different domain-specific modeling languages.

Complementary graph concept [6] (reviewed in Section 3) allows us to distribute a centralized graph without any modification of the graph grammar transformations rules that have been used to describe a given problem in the centralized case. The GRADIS multiagent framework (the acronym of GRAph DIStribution toolkit) offers the set of Local Graph Transformations Agents (LGTA),

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 280–288, 2010.
© Springer-Verlag Berlin Heidelberg 2010

distributed over the network, that are able to make the transformations on the local graphs and cooperate to achieve a common goal. The correctness of a distributed graph transformation system is guaranteed by assurance that after gluing of the partial graphs a resultant graph belongs to a family of graphs generated by the sequential graph transformation. In Section 4 the possibility of application of complementary graph concept to the Double Pushout approach is shown. In Section 5 we will focus on the proving the polynomial complexity of an algorithm of a coordination between the agents maintaining particular partial graphs in a distributed parallel agent environment and a correctness of this solution in the context of time dependent errors or deadlocks.

Polynomial computational complexity of the coordination algorithms and introduction of the parallelism into graph transformation mechanism improves the efficiency of algorithmic graph transformations and enables the specification of the agent systems and online control of their behavior.

2 Double Pushout Grammars

In this paper we will consider labeled (attributed) graphs. Let Σ^v and Σ^e be the sets of node and edge labels respectively. The graph structure is defined as follows:

Definition 1. *(Σ^v, Σ^e)-graph is a triple (V, E, φ) where V is nonempty set, E is a subset of $V \times \Sigma^e \times V$, and $\varphi : V \longrightarrow \Sigma^v$. We denote the family of (Σ^v, Σ^e)-graphs as $\mathcal{G}$.*

For any $G = (V, E, \varphi) \in \mathcal{G}$, V is set of nodes, E is set of edges and φ is node labeling function. One can extend this graph definition e.g. introducing attributing functions for both, nodes and edges, but these extensions will not influence the rules of the centralized graph distribution and its transformation. For that reason they will not be considered here.

The object of our interest are the double pushout graph grammars (DPO). DPO rule is denoted as $P : L \leftarrow K \rightarrow R$, $L \supseteq K \subseteq R$, where K is an interface graph, L and R graphs are the left and right hand side of the production respectively, $L, K, R \in \mathcal{G}$.

P application may be described informally in the following way. First one has to find a match of L, denoted as $m(L)$, in G. The match m is assumed to be a graph morphism preserving gluing condition consisting of:

- *Identification condition* - for two nodes, $x, y \in V(L)$, $m(x) = m(y)$ iff $x = y$ or $x, y \in V(L) \cap V(R)$.
- *Dangling condition* - $\forall e \in E(G - m(L))$, none of the e endpoints belongs to $m(L) - m(K)$.

After finding $m(L)$ in G one has to remove all the nodes of $m(L) - m(K)$ from G, with all the adjacent edges. Finally, all the nodes of $m(R) - m(K)$ together with the convenient edges have to be added to G.

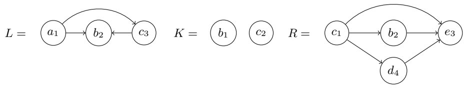


Fig. 1. Sample DPO production $P = (L \leftarrow K \rightarrow R)$ or $(L \supseteq K \subseteq R)$. Labels subscripts denote the local indices ascribed to the particular nodes.

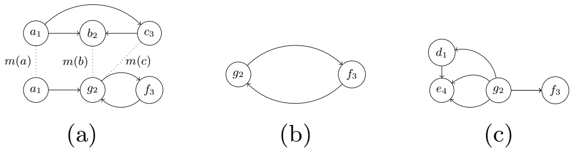


Fig. 2. Three steps of DPO production. (a) $m(L)$ match in G (bottom graph) according to the morphism m. Dotted lines show the $m : L \rightarrow G$: $m(b) = m(c) = g$, $m(a) = a$, (b) temporary graph $Z = G-(m(L)-m(K))$, (c) resultant graph $G' = Z \cup (m(R)-m(K))$.

Note that the dangling condition implies that all the edges belonging to $E(G - m(L))$ and being adjacent to $m(L)$ remain unchanged after applying DPO production. It impacts an agent incorporation strategy while applying a production in a distributed environment. The formal definition of DPO can be found in [2,4].

The following example illustrates the DPO grammar production. We deal with the graph G shown in Fig.2a (bottom) and the production P (see Fig.1) to be applied on.

The application of P is performed in three steps:

- a match of L in G (according to the morphism m) is found, in Fig.2a we have shown the morphism m mapping the nodes of L into the nodes of G: $m(a_1) = a_1$, $m(b_2) = m(c_3) = g_2$.
- the node $a_1 = m(L) - m(K)$ is removed from the graph G and the intermediate graph Z (Fig.2b) is obtained; note that the gluing condition is satisfied,
- the nodes and the corresponding edges from $m(R) - m(K)$ are added to the graph Z . The resultant graph G' is shown in Fig.2c.

In the paper we continue the consideration on distributing the algebraic graph grammar productions and propose a method of distributing double pushout grammars (DPO).[1]

3 Complementary Graphs

For DPO graph grammars the key step leading to distributed graph transformations is partitioning the centralized graph G into the set of subgraphs G_i, called *complementary graphs* and managed by the agents. Next, the complementary

[1] For the clarity we assume that morphism m is an identity map: $m(H) = H$. If needed, m is set explicitly.

graphs are distributed to different locations. Transformation of each subgraph G_i will be controlled by its Local Graph Transformation Agent (LGTA$_i$); in Section 4 we will describe a cooperation between LGTA's in DPO graph transformation systems.

To maintain the consistency between centralized graph G and the set of distributed graphs, some nodes (called border nodes) should be replicated and placed in the proper partial graphs. Graphically, we will mark a border node by a double circle.

We introduce following notations: $\mathsf{Border}(G_i)$ is a set of all border nodes of the graph G_i; the function $\mathsf{PathS}(G^\star, v, w)$ returns a set of the nodes belonging to the edges creating any acyclic connection between v and w in G; for a given graph H, $H^\star$ will denote an underlying undirected graph. During the splitting a graph we are interested in checking whether an edge connecting two nodes crosses a border among the subgraphs. For example, for the graph G presented in Fig.3a $\mathsf{PathS}(G^\star, 6, 8) = \{0, 1, \ldots 8\}$.

Definition 2. *A set of partial graphs $G_i = (V_i, E_i, \varphi_i), i = 1, 2, \ldots k$, is a complementary form of graph G iff there exists a set of injective homomorphisms $s_i : G_i \longrightarrow G$ such that:*

1. $\bigcup_{i=1,\ldots k} s_i(G_i) = G$
2. $\forall i, j \in \{1, \ldots k\} : s_i(V_i) \cap s_j(V_j) = s_i(\mathit{Border}(G_i)) \cap s_j(\mathit{Border}(G_j))$
3. $\forall w \in V_i \forall v \in V_j : \exists p = \mathit{PathS}(G, w, v) \Rightarrow \exists b \in \mathit{Border}(G_i) : s_i(b) \in p$
4. $\forall i \in \{1, \ldots k\} : v \in \mathit{Border}(G_i) \Leftrightarrow$
   $(\exists w \in G_i : w \text{ is connected with } v)$ *or* $G_i = \{v\}$

Partial graph G_i is also referred to as a complementary graph.

The above formal definition may be difficult to use in practical construction of the partial graphs. The effective algorithm for splitting a graph into the two partial graphs is presented in [6]. Splitting a graph G into the any number of new partial graphs can be accomplished by the recursive execution of this algorithm on the already obtained partial graphs H' or H'' and so on.

The example of three distributed partial graphs obtained from G shown in Fig.3a is presented in Fig. 3b [2].

For any border node v in graph G_i we can move boundary in such a way, that all nodes, that are connected with v (inside another complementary graphs) are incorporated to G_i as border nodes and replicas of v are removed from another graphs (i.e. v becomes a normal node); we will do it using $\mathsf{Incorporate}(v, i)$ operation. In Figures 3b and 4 the partial graphs before and after $\mathsf{Incorporate}((-1, 1), 2)$, $\mathsf{Incorporate}((-1, 2), 2)$, $\mathsf{Incorporate}((-1, 3), 2)$ are presented.

[2] Let's note that a node index consists of two numbers: the first one denotes an ordinal number of a given partial graph or equals to -1 in a border node case and the second one is an unique index inside a particular partial graph or, for border nodes, a unique index in the set of all border nodes. All replicas of a given border node have identical index of the form $(-1, j)$.

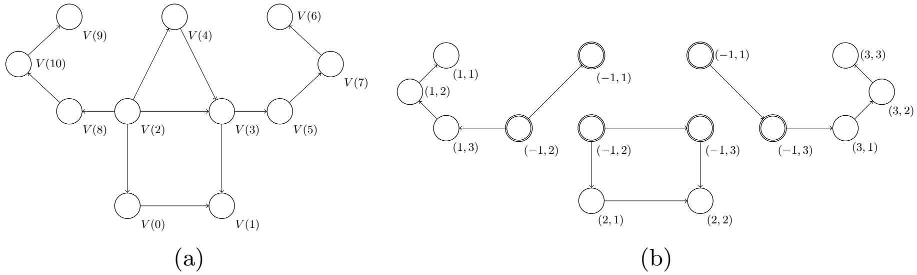


Fig. 3. (a) G in the initial, centralized form (b) Complementary form of G, $\{G_1, G_2, G_3\}$

4 Distributed Graph Maintenance

In our approach each of the partial graphs is managed by its agent. We assume that the agent X_0 applies the DPO grammar production $P : L \leftarrow K \rightarrow R$ on its partial graph G_0 being the subset of distributed form of the centralized graph G. To apply the production P, X_0 has to follow four steps: (*1*) determine $V(L)$, (*2*) find an occurrence of $V(L)$ in the distributed environment, (*3*) incorporate $V(L)$, (*4*) apply locally production P.

First step is finding an occurrence of L in the partial graph managed by the agent X_0 and/or, in the neighbor ones. X_0 does that starting from some *seed* node $v \in V(L)$ and making the lookup in a neighborhood of v. It's obvious that $L \subseteq B = k - \mathsf{neighborhood}(v)$ set, where for a given L, $k \leq V(L)$ is a diameter of an underlying undirected graph $L^\star$.

Remark 1. For sufficiently high k, there may exist a partial graph, say G_i such that $G_i \subsetneq k - \mathsf{neighborhood}(v_0)$. In such a case X_i agent will rely the X_0 request, to the G_i neighbor graphs (agents) to serve X_0 query.

As the nodes of B set can be placed in other partial graphs B is computed in cooperation with the other agents: X_0 sends a request to those agents (which are denoted as $X_i, i = 1, 2, \dots n$) and receives required data $(B_i, i = 1, 2, \dots n)$ in a feedback. It has to remarked that this data is volatile, i.e. some nodes may be removed from a given partial graph after delivering requested data to X_0. B is recovered form all B_i ($i = 0, 1, \dots n$, B_0 is X_0 own data), $B = \bigcup_i B_i$, where i enumerates subsequent agents. When an occurrence of L in B is found the X_0 may proceed to the second step.

The part of B which is essential for the production P is $B' = L \cap B$. At the level of particular partial graphs it is $B'_i = L \cap B_i$.

In the second step X_0 has to ensure that L will be accessible for applying the production P. According to the two-phase commit protocol semantics, in the first phase the agent X_0 sends a request to the other agents, X_i, to block all nodes belonging to corresponding B'_i. X_i blocks the nodes (`agreement=yes`) or rejects the request in two possible circumstances:

1. some nodes of B'_i are already blocked (`agreement=noaccess`) or
2. some nodes of B'_i don't exist in a partial graph managed by X_i (`agreement=nonexist`), e.g. they were just incorporated by some other agents.

Second phase depends on a feedback data received by X_0. It can follow one of three possible scenarios:

1. If all responds are `agreement=yes` then X_0 sends `commit` request to all X_i which supply corresponding $B''_i = B'_i \cap V(L)$ to X_0 in respond.
2. At least one respond is `agreement=nonexist` then X_0 sends `abort` request to all X_i and tries to find B (B') again.
3. At least one respond is `agreement=noaccess` then X_0 sends `abort` request to all X_i and repeats first phase with random delay.

In the third step, when all responds are `agreement=yes`, X_0 shifts local graphs boundaries (by incorporating corresponding B''_i subgraphs) in such a way that he gets able to apply production P locally. Note that after incorporating $B''_i \quad (i = 1, 2, \ldots k)$ by X_0, all nodes belonging to those subgraphs get *internal* nodes of G_0 (i.e. they are not border ones) and all nodes being the neighbor ones for B''_i get ore remain the border nodes for G_0.

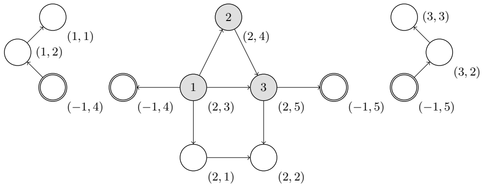


Fig. 4. Complementary form of G, $\{G'_1, G'_2, G'_3\}$, before local production $P = (L, K, R)$ (Fig.5a) on G'_2. All nodes matching L have been incorporated to G'_2 (gray shaded).

Let's consider the DPO graph grammar production $P : L \leftarrow K \rightarrow R$ (Fig.5a) to be applied on complementary form of the given centralized graph G shown in Fig.3a. We want to perform this production in the distributed environment consisting of three complementary graphs as shown in Fig.3b. The agent maintaining complementary graph G_2, say A_2, is to apply P locally. The first step A_2 has to make is to assure an exclusive access to $V(L)$ matching nodes by incorporating them. The detailed description of it is presented below.

The initiating agent A_2 managing the partial graph G_2 has to discover and incorporate all the nodes of L. The border nodes $(-1, 1), (-1, 2)$ are already present in G_2 (they matches the nodes of L indexed with 1 and 3 as shown in Fig.5a). Now A_2 tries to discover the lacking match of the node of L, indexed with 2. $L^\star$ diameter is 1 hence A_2 sends the request to each partial graph containing a replica of $u = (-1, 1)$ or $w = (-1, 2)$ to supply $B_i = 1 - \mathsf{neighborhood}(v_b)$, where $v_b = u$, w respectively. Table 1 presents the set of nodes of the particular

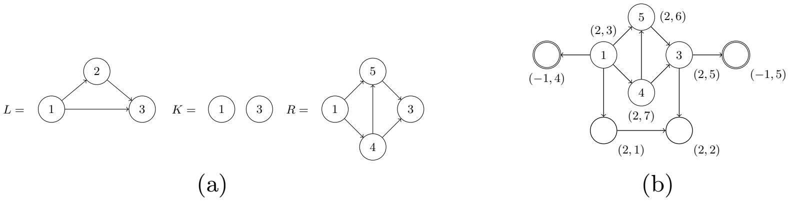


Fig. 5. (a) The sample DPO production (L, K, R). (b) Complementary form of G''_2 obtained after local production P (Fig.5a) on G'_2.

Table 1. The B_i and B'_i sets

i	B_i	B'_i
$i = 1$ (req. to A_1)	$(-1, 1), (-1, 2), (1, 3)$	$(-1, 1), (-1, 2)$
$i = 2$ (req. to A_3)	$(-1, 1), (-1, 2), (3, 1)$	$(-1, 1), (-1, 2)$

B_i and B'_i, $(i = 1, 2)$. Using B'_is instead of B_is may improve effectiveness by reducing the number of the nodes to be incorporated.

After identifying an occurrence of L in $B' = \bigcup_{i=1}^{3} B'_i$, A_2 incorporates all nodes matching the L, namely $(-1, 1)$, $(-1, 2)$, $(-1, 3)$. We assume that no parallel action is made by A_1 and A_3 agents so A_2 receives `agreement=yes` messages and sends the `commit` request to them. After subsequent incorporations of the nodes $(-1, 2), (-1, 1), (-1, 3)$ (their new indexation is $(2, 3), (2, 4), (2, 5)$ respectively) we obtain G'_2. The left hand side graph L can be matched appropriately to the internal nodes $1, 2, 3$ of G'_2 (see Fig.4) and thus the production P can be applied locally. Resultant complementary graph G''_2 is shown in Fig.5b.

5 Effectiveness Issues and Conclusions

The effectiveness of the graph transformation formalism in practical application is limited due to its high computational complexity of parsing and membership problems. This effectiveness can be improved by introduction the parallel execution paradigm. One of the more effective approaches is to distribute automatically the centralized graph and execute given algorithms parallely on its local parts; such an approach is very convenient for a system designer which has not to take care about the problems with the parallelism and implicit synchronization. GRADIS agent framework supports such an approach by introducing the complementary graphs concept and the agent framework supporting cooperation in the complementary graph modifications.

Let's consider the complexity of determination of the graph B defined as $B = \mathsf{neighborhood}(v_0)$. Formally we have to visit all the direct neighbors of v_0 and mark them as belonging to the *layer 1*; next, for each a node belonging to *layer j* $(j < k)$ we have to visit those of their direct neighbors that don't belong to the lower layers; we stop this algorithm when either we have determined the members of the k-th layer or all the nodes of the considered graph are already marked. Let's note that this algorithm excludes cycles and backtracking in nodes visiting and its worst complexity is $\mathcal{O}(n)$, where n is number of nodes in the centralized graph. The estimation of a number of messages which are necessary to assemble a distributed information is more difficult. The agent X_0 sends, in the worse case, the messages to all other agents maintaining partial graphs (we assume that a request of coordination between a several boundary nodes belonging to G_0 and some G_i can be packed in a single message); unfortunately (with respect to the remark 1) some agent, say X_u, maintaining the node u being a direct neighbor of v_0 and belonging to other partial graph, may also send a message to all other agents (except of those that maintain the paths between v_0 and this node). Thus X_u may generate a duplicate of the request sent previously by X_0. Following this reasoning, for a number of agents equaled to p the upper limitation of the number of sent messages is $p(p-1)\dots(p-k+1) \in \mathcal{O}(p^k)$. In a case when some node v receives several duplicated requests attaching it to B, only first is serviced. All subsequent duplicated messages are ignored. v can receive not more then next $p-2$ messages from the other agents (assuming that an agent remembers its local activity and does not send a message to the same external node twice) but this doesn't trigger any additional activity so the limitation for a number of the sent messages is $n \cdot p$. We consider both of these limitations because in practice they help to reduce number of sent messages. The obtained B_i graphs are returned directly to X_0 agent so both actions, prepare or commit/abort phases, are limited to $O(p)$ sent messages.

The efficiency of the system will depend, however, on a final degree and quality of the considered problem decomposition. The earlier works with the algorithmical graph transformations [7,8] create good prognosis for this.

References

1. Baland, P., Corradini, A., Montanari, U., Rossi, F.: Concurrent sematics of algebraic graph transformations, pp. 107–187 (1999)
2. Corradini, A., Montanari, U., Rossi, F., Ehrig, H., Heckel, R., Löwe, M.: Algebraic approaches to graph transformation - part i: Basic concepts and double pushout approach. In: Handbook of Graph Grammars and Computing by Graph Transformations. Foundations, vol. 1, pp. 163–246. World Scientific, Singapore (1997)
3. Ehrig, H., Heckel, R., Lowe, M., Ribeiro, L., Wagner, A.: Algebraic Approaches to Graph Transformation – Part II: Single Pushout and Comparison with Double Pushout Approach, pp. 247–312
4. Ehrig, H., Ehrig, K., Prange, U., Taentzer, G.: Fundamentals of Algebraic Graph Transformation. In: Monographs in Theoretical Computer Science. An EATCS Series. Springer, New York (2006)
5. Ehrig, H., Ermel, C., Hermann, F.: On the relationship of model transformations based on triple and plain graph grammars. In: GRaMoT 2008: Proceedings of the Third International Workshop on Graph and Model Transformations, pp. 9–16. ACM, New York (2008)

6. Kotulski, L.: GRADIS – Multiagent Environment Supporting Distributed Graph Transformations. In: Bubak, M., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2008, Part III. LNCS, vol. 5103, pp. 644–653. Springer, Heidelberg (2008)
7. Kotulski, L., Sędziwy, A.: Agent Framework For Decomposing a Graph Into the Equally Sized Subgraphs. In: WORLDCOMP 2008 Conference, Foundations of Computer Science, pp. 245–250 (2008)
8. Kotulski, L., Strug, B.: Parallel Graph Transformation in Adaptive Design. In: Second Intenational Workshop Graph Computation Models Leicester 2008, pp. 43–50 (2008)
9. Montanari, U., Pistore, M., Rossi, F.: Modeling concurrent, mobile and coordinated systems via graph transformations, pp. 189–268 (1999)
10. Morandini, M., Penserini, L., Perini, A.: Operational semantics of goal models in adaptive agents. In: AAMAS 2009: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems, Richland, SC, pp. 129–136. International Foundation for Autonomous Agents and Multiagent Systems (2009)
11. Peng, W., Krueger, W., Grushin, A., Carlos, P., Manikonda, V., Santos, M.: Graph-based methods for the analysis of large-scale multiagent systems. In: AAMAS 2009: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems, Richland, SC, pp. 545–552. International Foundation for Autonomous Agents and Multiagent Systems (2009)
12. Sycara, K.P.: Multiagent system. AI Magazine, 79–92 (1998)

Ant Agents with Distributed Knowledge Applied to Adaptive Control of a Nonstationary Traffic in Ad-Hoc Networks

Michal Kudelski[1,2] and Andrzej Pacut[1,2]

[1] Institute of Control and Computation Engineering, Warsaw Univ. of Technology
Nowowiejska 15/19, 00-665 Warsaw, Poland
[2] NASK Wawozowa 18, 02-796 Warsaw, Poland
m.kudelski@elka.pw.edu.pl, A.Pacut@ia.pw.edu.pl

Abstract. We analyze a SWARM-based multi agent control scheme for controlling the traffic of data packets in ad-hoc networks. We consider nonstationary traffic patterns. We demonstrate how the distributed and geographically localized knowledge gathered by ant agents may improve the effectiveness of the ant learning mechanism. Our experiments indicate the improvement of adaptation capabilities of ants under dynamic topology changes and dynamic load level changes in the network.

1 Introduction

An ad-hoc network is a set of wireless mobile nodes that dynamically form a temporary network without using any existing infrastructure. Each node of such network is autonomous and self-configurable. Typically, the communication between nodes is indirect: each node can also act as a wireless router and can be used as an intermediate node on the path from a source to a destination.

Ad-hoc networks may be also considered as dynamic systems. On the network level, the problem of routing may be defined as a problem of controlling the traffic of data packets. The issue is not trivial due to multiple challenges. Probably the most important one is the dynamics of topology changes: the routing policy needs to be continuously adjusted to dynamically changing connections in the network. The difficulty may be deepened by the nonstationary traffic pattern, i.e. the network's load level changing in time.

Examples of traffic patterns with a dynamically changing load level may include the Internet traffic (with periodical changes of users' activity) and variable bitrate (*VBR*) multimedia transmissions [14]. These patterns may be characterized by different time scales: the period of changes may vary between days, hours (Internet traffic) and fractions of seconds (VBR traffic).

In order to handle such a demanding nonstationarity, specialized routing control systems are required. Multi-agent ant learning mechanisms have already proved to be effective in solving the analogous problem in fixed telecommunication networks [11]. The aim of this paper is to show how ant agents deal

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 289–296, 2010.
© Springer-Verlag Berlin Heidelberg 2010

with nonstationary traffic in ad-hoc networks. We also demonstrate how the distributed geographical localization of knowledge [9] may improve the adaptation abilities of ant routing in ad-hoc networks. We analyze the adaptation abilities of a routing control mechanism by examining the network's response to dynamic changes of load level.

There have been many routing approaches emerging for ad-hoc networks recently [1,4,13]. There are also multiple approaches that base on learning mechanisms [8]. Amongst them, a group of ant routing algorithms may be identified [3,5,12]. However, in the literature there exists no works that focus on ad-hoc networks' response to nonstationary traffic patterns. Authors of [11] perform a similar analysis in fixed telecommunication networks.

2 Routing as a Problem of Control

From the control theory perspective, routing may be seen as a problem of control. A computer network may be seen as a dynamic system and a routing mechanism as a controller. Figure 1 shows the simplified scheme of a control loop for the problem of routing.

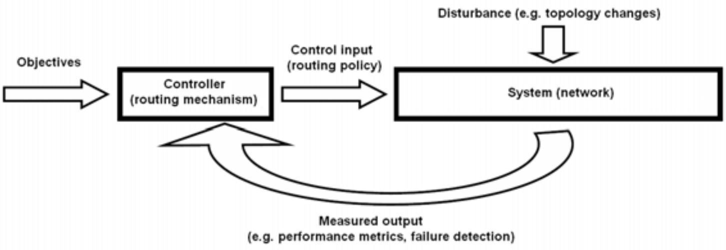


Fig. 1. Routing as a control problem)

The **controlled system** (network) is a complex dynamic system of various parameters. The current network state is determined by the network's load level, available connections between nodes, resources available in particular nodes, bandwidth of links, etc. Moreover, the network is exposed to multiple **disturbances** such as frequent topology changes, a noise affecting wireless links, link failures, etc.

A controller (routing mechanism) takes into account certain **objectives**. These objectives may include maximizing network performance from the application point of view (application requirements) while minimizing the cost of network itself in accordance with its capacity [2]. Some examples of performance requirements are hop count, end-to-end delay, delivery rate, throughput, jitter, etc. The network capacity is a function of available resources at each node, node density, network's load level and the frequency of topology changes.

At the same time, the routing mechanism **measures output** of the network. Depending on the objectives, various performance parameters may be measured and numerous performance metrics may be constructed. In addition, the routing mechanism may detect failures in the network (e.g. link failures) and attempt to perform recovery procedures.

Basing on objectives and the measured network output, the routing mechanism provides a **control input** to the network. The control input is determined be the *routing policy*, which is typically represented by routing tables. The routing policy may by updated periodically or even continuously (in *proactive routing*) or in a response to particular events taking place in the network (in *reactive routing*). These updates rely on finding the shortest paths to destinations (in a sense of the given metric). The algorithms used for finding the shortest paths are individual for particular routing mechanisms.

The problem of routing control may solved by *centralized routing mechanisms* or by *distributed routing mechanisms.* In centralized routing mechanisms, there is a central controller gathering information from the whole network and updating routing tables in all nodes. However, most of the routing mechanisms are distributed. In distributed mechanisms, nodes individually perform measurements, construct their routing tables and exchange the routing information with other nodes in the network. These operations may be carried out by multi-agent systems in general and by ant agents in particular.

3 Ant Agents with Distributed Knowledge

Ant agents employed for routing. AntHocNet, proposed in 2005 by Di Caro et al. [3], is an adaptive ant routing algorithm for ad-hoc networks. It is an example of a SWARM-based multi-agent approach: numerous types of ant agents are used to gather the routing information and exchange it between the nodes. *Reactive ants* are used to discover and set up new paths. *Proactive ants* are used to monitor the existing paths and explore the network for new paths. *Failure notification ants* are used to propagate information about failures through the network. Finally, *Repair ants* are employed to restore broken connections.

Geographical localization of knowledge. In order to improve the operation of ant agents, the *Geographical Localization of Knowledge* (GLK) was introduced in [7]. In the proposed approach, the routing information gathered by learning agents (*the knowledge*) is connected with geographical locations in the network rather than with individual nodes. Thus, the information about connections in the network may be defined on a *higher* locations' level rather than on a *lower* nodes' level. Such knowledge is then *logically* stored within geographical locations and exchanged between the nodes as they change their positions. The model of connections on locations' level turned out to be more stable than the model on nodes' level, i.e. connections on nodes' level are broken much more frequently than connections on locations' level.

Distributing the knowledge between agents. *Distributed Geographical Localization of Knowledge* [9] provides the practical realization of the GLK approach. Knowledge is logically stored in geographical locations yet it is physically distributed between the nodes within particular locations. *The knowledge exchange mechanism* is provided that allows for distributing and synchronizing the information between the nodes.

AntHocGeo, proposed and discussed in detail in [7,9], is a modification of AntHocNet that implements the concept of distributed geographical localization of knowledge. Equivalent agents are employed as in AntHocNet, yet they operate on a distributed and localized knowledge.

In this paper, we analyze and compare the operation of AntHocNet and AntHocGeo in networks with dynamically changing load level. We focus on the influence of applying distributed knowledge on the adaptation capabilities of ant agents.

4 Experimental Results

All experiments were performed using ns2 simulation environment [10]. We analyze the "response" of the network controlled by a routing mechanism for the following load level changes: unit step (both jump and drop of the network load level), periodical changes of the load level in the form of a sine wave and periodical changes in the form of a rectangular wave.

The results are presented on figures consisting of three charts. The first chart shows the average end-to-end delay as a function of time. The second chart shows the number of generated bytes of data and the number of delivered bytes of data (in the whole network) as functions of time for AntHocGeo algorithm (distributed implementation [9], basic parameters as in [7]). The last chart shows the same as the second chart yet for AntHocNet algorithm (parameters as in [3]). All the presented values are averaged in a moving window (window size is set to 100 seconds) and in addition, an exponential smoothing with the α-parameter equal to 0.9 is performed.

We use two communication scenarios: an *ad-hoc network of mobile robots* inside a building (same as in [6]) and an *outdoor ad-hoc network* with nodes placed on a plane without obstacles (1000×1000 m^2). We use the *Random Waypoint* mobility model.

4.1 Unit Step

The unit step scenario of traffic is realized as follows. There are 30 mobile robots in the network and there are 5 CBR sources, each of them generating one 64-byte packet per second during the whole simulation (starting at the simulation time of 15.0 seconds). At the time of 100.0 four additional CBR sources start generating data: each additional data source generates 50 packets of the same size per second. At the time of 150.0 additional sources stop their activity. Hence, we have a sudden jump of the load level at 100.0 and a sudden drop of the load level at 150.0. The results are presented on Fig. 2.

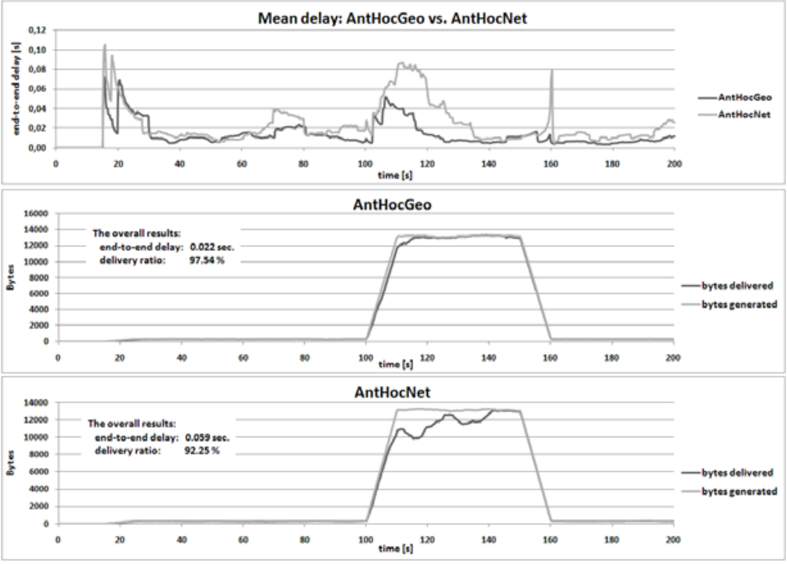


Fig. 2. Unit step: averaged end-to-end delay(top), averaged number of bytes generated and delivered for AntHocGeo (middle) and AntHocNet (bottom) algorithms. Mobile robots environment [6], nodes' speed: 3 m/s.

The average end-to-end delay as a function of time indicates that the learning mechanism of AntHocGeo achieves lower delays and it adapts faster than the corresponding mechanism of AntHocNet. It can be clearly observed both during a sudden jump and a sudden drop of the load level. The peak corresponding to the jump is much higher and much wider in the case of AntHocNet. Moreover, there is no significant peak corresponding to the sudden drop of the load level in the case of AntHocGeo, while there is one in the case of AntHocNet.

The same observations may be made when observing the number of bytes generated and delivered in the network. The time of adaptation to a new load level is much shorter in the network controlled by AntHocGeo: the plot of the number of delivered bytes fits to the plot of generated bytes after no more than 5 seconds (whereas AntHocNet needs more than 30 seconds). The overall results validate the superiority of AntHocGeo.

The results obtained for a network of mobile robots indicate that the localized and distributed knowledge improves the adaptation of ant agents to a sudden jump and a sudden drop of the load level in mobile ad-hoc networks.

4.2 Sine Wave

The sine wave scenario of traffic is realized as follows. There are 60 nodes in the network and there are 5 CBR sources. Each source starts generating 64-byte data packets at the simulation time of 15.0 sec., with the CBR interval of 0.25 seconds (4 packets generated per second). At 35.0 the interval of each data source starts changing periodically. The values change within the range of 0.16 and 0.5 sec. Periodical changes have the form of a sine wave with the period set to 60 seconds. The results are showed on Fig. 3.

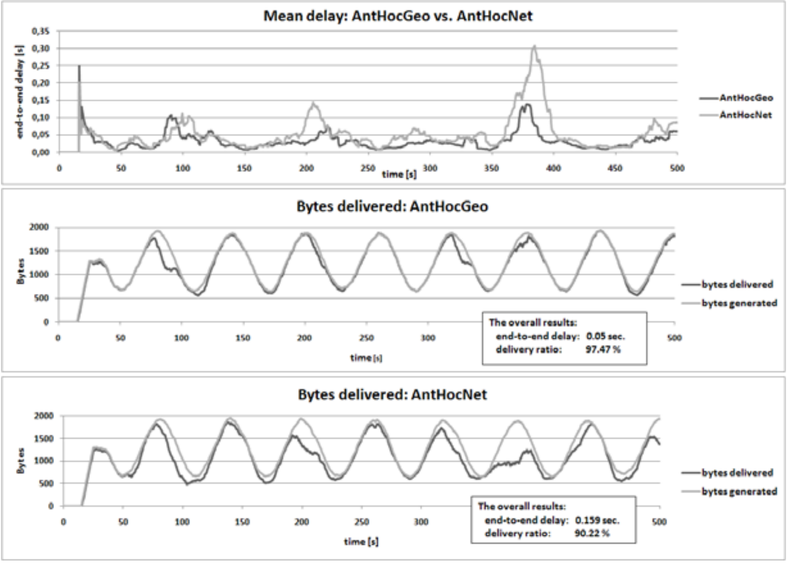


Fig. 3. Sine wave: averaged end-to-end delay(top), averaged number of bytes generated and delivered for AntHocGeo (middle) and AntHocNet (bottom) algorithms. Outdoor ad-hoc network, nodes' speed: 20 m/s.

Again, the overall results demonstrate the advantage of AntHocGeo: the end-to-end delay achieved by AntHocNet is more than three times as big as the one achieved by AntHocGeo. At the same time, AntHocGeo achieves a higher delivery ratio.

The adaptation capabilities of the approach with distributed GLK are much better as well. Looking at the average end-to-end delay as a function of time, it may be observed that there are some significant peaks of delays. The peaks may occur when the load level grows and they may be additionally intensified by temporary lacks of connectivity in the network. The peaks are correlated with local disturbances in the number of delivered bytes (more data packets are lost). One may notice that AntHocGeo adapts very well to the periodical changes of the load level — the plot of the number of delivered bytes fits very well to the plot of the number of bytes generated in the network. Furthermore, AntHocGeo adapts better to dynamic changes in the topology of the network: the end-to-end delay peaks are lower than in the case of AntHocNet and much fewer data packets are lost due to the temporary lack of connectivity in the network.

4.3 Rectangular Wave

The rectangular wave scenario of traffic is realized as follows. There are 60 nodes in the network and there are 5 CBR sources, each of them generating one 64-byte packet per second during the whole simulation (starting at the simulation time of 15.0 seconds). At 100.0 four additional sources start generating an additional traffic in the form of a rectangular wave. Additional sources switch their interval parameter between two values: 0.1 and 2.0 seconds. The period of the rectangular wave is set to 50 seconds. The results are showed on Fig. 4.

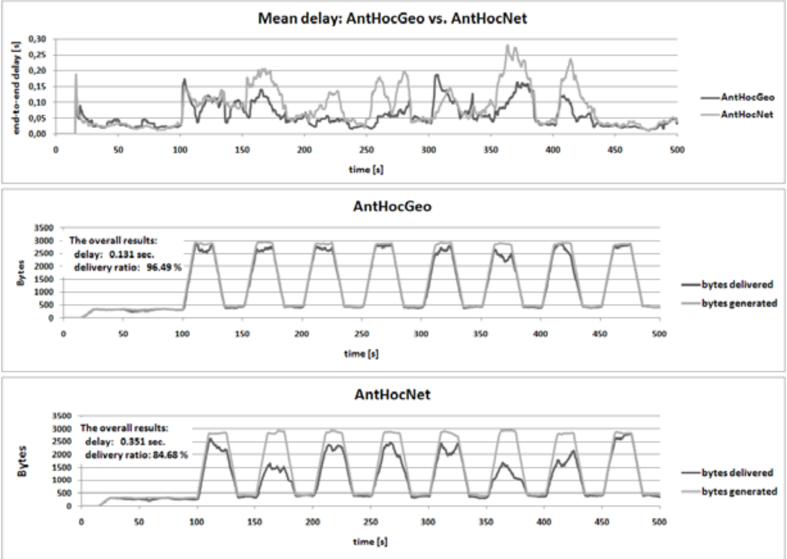


Fig. 4. Rectangular wave: averaged end-to-end delay(top), averaged number of bytes generated and delivered for AntHocGeo (middle) and AntHocNet (bottom) algorithms. Outdoor ad-hoc network, nodes' speed: 20 m/s.

The output of the experiment is very similar to the output of the experiment with load level in the form of a sine wave. The overall results achieved by AntHocGeo outperform the ones obtained by AntHocNet. At the same time, the adaptation capabilities of the learning mechanism are improved by applying the higher-level model of connections in the network.

5 Conclusions

The problem of controlling a nonstationary traffic in ad-hoc networks is a very challenging task. The control scheme employing ant agents allows to solve it in a satisfying way. The adaptation capabilities of the ant learning mechanism may be further improved by applying the distributed geographical localization of knowledge. The localized and distributed knowledge gathered by ant agents provides the learning mechanism with a higher level model of connections and improves the network's response to dynamic topology changes and dynamic load level changes.

Acknowledgments. The scientific work funded as a research project by finance for scientific research in the years 2009-2010.

References

1. Abolhasan, M., Wysocki, T., Dutkiewicz, E.: A review of routing protocols for mobile ad hoc networks. Ad Hoc Networks 2(1), 1–22 (2004)
2. El-Nabiali, T.H.A., Ahmed, A.: Modeling and simulation of a routing protocol for ad-hoc networks combining queuing network analysis and ant colony algorithms. PhD thesis, Universität Duisburg-Essen (2005)
3. Di Caro, G., Ducatelle, F., Gambardella, L.M.: Anthocnet: An adaptive nature-inspired algorithm for routing in mobile ad hoc networks. European Transactions on Telecommunications 16, 443–455 (2005)
4. Hong, X., Xu, K., Gerla, M., Angeles, L.C.A.: Scalable routing protocols for mobile ad hoc networks. IEEE network 16(4), 11–21 (2002)
5. Kalaavathi, B., Madhavi, S., VijayaRagavan, S., Duraiswamy, K.: Review of ant based routing protocols for manet. In: International Conference on Computing, Communication and Networking, ICCCN 2008, December 2008, pp. 1–9 (2008)
6. Kudelski, M., Gadomska-Kudelska, M., Pacut, A.: Geographical cells routing in ad-hoc networks of mobile robots. In: The 14th IEEE Mediterranean Electrotechnical Conference, MELECON 2008, May 2008, pp. 374–379 (2008)
7. Kudelski, M., Pacut, A.: Geographical cells: Location-aware adaptive routing scheme for ad-hoc networks. In: EUROCON, 2007. The International Conference on "Computer as a Tool", September 2007, pp. 649–656 (2007)
8. Kudelski, M., Pacut, A.: Learning methods in ad-hoc networks: a review. Evolutionary Computation and Global Optimization, Prace Naukowe Politechniki Warszawskiej, Elektronika z. 160, 153–163 (2007)
9. Kudelski, M., Pacut, A.: Ant routing with distributed geographical localization of knowledge in ad-hoc networks. In: EvoWorkshops 2009: Proceedings of the EvoWorkshops 2009 on Applications of Evolutionary Computing, pp. 111–116. Springer, Heidelberg (2009)
10. ns2. The network simulator, http://www.isi.edu/nsnam/ns
11. Pacut, A., Gadomska-Kudelska, M., Igielski, A.: Ant-routing vs. q-routing in telecommunication networks. In: 20th European Conference on Modelling and Simulation, ECMS, pp. 67–72 (2006)
12. Rajagopalan, S., Shen, C.-C.: Ansi: a swarm intelligence-based unicast routing protocol for hybrid ad hoc networks. J. Syst. Archit. 52(8), 485–504 (2006)
13. Royer, E.M., Toh, C.-K.: A review of current routing protocols for ad hoc mobile wireless networks (1999)
14. Won, Y., Ahn, S.: Gop arima: Modeling the nonstationarity of vbr processes. Multimedia Systems 10(5), 359–378 (2005)

Dynamic Matrix Control Algorithm Based on Interpolated Step Response Neural Models

Maciej Ławryńczuk

Institute of Control and Computation Engineering, Warsaw University of Technology
ul. Nowowiejska 15/19, 00-665 Warsaw, Poland, tel. +48 22 234-76-73
M.Lawrynczuk@ia.pw.edu.pl

Abstract. This paper presents a nonlinear Dynamic Matrix Control (DMC) algorithm. A neural network calculates on-line step response coefficients which comprise a model of the controlled process. These coefficients are next used to determine the optimal control policy from an easy to solve quadratic programming problem. To reduce the number of model parameters (step response models usually need many coefficients) interpolated step response neural models are used in which selected coefficients are actually calculated by the neural network whereas remaining ones are interpolated by means of cubic splines. The main advantage of the step response neural model is the fact that it can be obtained in a straightforward way, no recurrent training is necessary. Advantages of the described DMC algorithm are: no on-line model linearisation, low computational complexity and good control accuracy.

Keywords: Process control, Dynamic Matrix Control, neural networks, interpolation, optimisation, quadratic programming.

1 Introduction

Model Predictive Control (MPC) algorithms [7,13,14] have been successfully used for years in many industrial applications [12]. It is because, unlike other control techniques, in a natural way they can take into account constraints imposed on process inputs (manipulated variables) and outputs (controlled variables), which usually decide on quality, economic efficiency and safety. Moreover, MPC techniques are very efficient in multivariable process control.

Dynamic Matrix Control (DMC) [2] algorithm is used in the majority of industrial applications [1,12,13,14]. It uses step response models which can be obtained in a straightforward way. It is particularly important in the industry. To find a model, it is only necessary to record the output response of the process to a step excitation signal, no complicated model identification algorithms are used. In many cases accuracy of the classical DMC algorithm is very good, in particular when set-point changes are not big and frequent. On the other hand, when applied for significantly nonlinear processes the DMC algorithm based on linear step response models may be insufficiently accurate, even unstable.

Over the last years a number of nonlinear MPC algorithms have been developed [4,14]. They use different model structures, for example fuzzy models

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 297–304, 2010.
© Springer-Verlag Berlin Heidelberg 2010

[9,14], Volterra models [8] and artificial neural networks [6,10,14]. In particular, neural models can be efficiently used on-line in MPC because they have excellent approximation abilities, a small number of parameters and a simple structure.

This paper is concerned with a nonlinear DMC algorithm presented originally in [5]. For prediction step response neural models are used, i.e. for the current operating point step response coefficients are calculated on-line by a neural network. Because step response models usually need many coefficients, in this work interpolated step response neural models are considered. In this approach only selected coefficients are actually calculated by the neural network whereas remaining ones are interpolated by means of cubic splines.

2 Model Predictive Control Algorithms

In MPC algorithms [7,13,14] at each consecutive sampling instant k, $k = 0, 1, \ldots$, a set of future control increments is calculated

$$\triangle \boldsymbol{u}(k) = [\triangle u(k|k)\ \triangle u(k+1|k) \ldots \triangle u(k+N_{\mathrm{u}}-1|k)]^{\mathrm{T}} \tag{1}$$

It is assumed that $\triangle u(k+p|k) = 0$ for $p \geq N_{\mathrm{u}}$, where N_{u} is the control horizon. The objective of MPC is to minimise differences between the reference trajectory $y^{\mathrm{ref}}(k+p|k)$ and predicted outputs values $\hat{y}(k+p|k)$ over the prediction horizon $N \geq N_{\mathrm{u}}$, i.e. for $p = 1, \ldots, N$. For optimisation of the future control policy (1) the following quadratic cost function is usually used

$$J(k) = \sum_{p=1}^{N} (y^{\mathrm{ref}}(k+p|k) - \hat{y}(k+p|k))^2 + \sum_{p=0}^{N_{\mathrm{u}}-1} \lambda_p (\triangle u(k+p|k))^2 \tag{2}$$

where $\lambda_p > 0$ are weighting coefficients. Only the first element of the determined sequence (1) is actually applied to the process, i.e. $u(k) = \triangle u(k|k) + u(k-1)$. At the next sampling instant, $k+1$, the prediction is shifted one step forward and the whole procedure is repeated.

Usually, constraints must be imposed on input and output variables. In such a case future control increments are found on-line from the following optimisation problem (for simplicity of presentation hard output constraints [7,14] are used)

$$\begin{aligned} &\min_{\triangle u(k|k), \ldots, \triangle u(k+N_u-1|k)} \{J(k)\} \\ &\text{subject to} \\ &\quad u^{\min} \leq u(k+p|k) \leq u^{\max}, \quad p = 0, \ldots, N_{\mathrm{u}}-1 \\ &\quad -\triangle u^{\max} \leq \triangle u(k+p|k) \leq \triangle u^{\max}, \quad p = 0, \ldots, N_{\mathrm{u}}-1 \\ &\quad y^{\min} \leq \hat{y}(k+p|k) \leq y^{\max}, \quad p = 1, \ldots, N \end{aligned} \tag{3}$$

where $u^{\min}$, $u^{\max}$, $\triangle u^{\max}$, $y^{\min}$, $y^{\max}$ define constraints. Predictions $\hat{y}(k+p|k)$ of the output variable over the prediction horizon, i.e. for $p = 1, \ldots, N$, are calculated by means of a dynamic model of the process.

In the classical DMC algorithm properties of the process are modelled by the discrete step response model which shows reaction of the process output to a step excitation input signal, usually a unit step. The output of the classical linear step response model is [14]

$$y(k) = y(0) + \sum_{j=1}^{k} s_j \triangle u(k-j) \tag{4}$$

Real numbers $s_1, s_2, s_3, \ldots$ are step response coefficients of the model. Assuming that the process is stable, after a step change in the input the output stabilises at a certain value s_∞, i.e. $\lim_{k\to\infty} s_k = s_\infty$. Hence, the model needs only a finite number of step response coefficients: $s_1, s_2, s_3, \ldots, s_D$, where D is named a horizon of the process dynamics.

3 Dynamic Matrix Control Based on Interpolated Step Response Neural Models

3.1 Step Response Neural Models

In the neural DMC algorithm [5] properties of the process are described by a step response neural model. Intuitively, the idea is straightforward: instead of constant step response coefficients $s_1, s_2, s_3, \ldots, s_D$, the model is comprised of coefficients $s_1(k), s_2(k), s_3(k), \ldots, s_D(k)$ whose values depend on the current operating point of the process determined by most recent measurements, i.e. the previous input value and the current output value

$$s_p(k) = f_p(u(k-1), y(k)) \tag{5}$$

where $p = 1, \ldots D$, functions $f_p \colon \mathbb{R}^2 \to \mathbb{R}^D$ are realised by a neural network. Analogously to the classical linear step response model (4), its output is

$$y(k) = y(0) + \sum_{j=1}^{k} s_j(k) \triangle u(k-j) \tag{6}$$

Thanks to excellent approximation properties, a Multi Layer Perceptron (MLP) neural network [3] can be used to calculate step response coefficients for the current operating point as discussed in [5]. The network has 2 input nodes, one hidden layer and D linear outputs. If necessary, the model also takes into account a measured disturbance (the uncontrolled input) [5].

Neural step response models are obtained in a straightforward way, no recurrent training algorithms are necessary. First, step responses obtained for different operating conditions must be recorded. Next, the neural network is trained off-line to find relations (5) between previous input and output values $(u(k-1), y(k))$ and step response coefficients $s_1(k), \ldots, s_D(k)$. If a unit step excitation signal cannot be used, step response coefficients should be scaled [5]. Unlike input-output or state-space models, the step response neural model is not used recurrently for prediction. Thanks to it, the prediction error is not propagated.

3.2 Interpolated Step Response Neural Models

Because step response models usually need many coefficients (typically a few dozens), one may reduce model complexity by using interpolated step response neural models. In this approach only selected step response coefficients are actually calculated by the neural network whereas remaining ones are interpolated by means of cubic splines.

Let the set P contain indices of coefficients which are actually calculated by the neural network and let the number of these coefficients be $\widetilde{D}$. For example, for $P = \{1, 5, 10, 25, 40, 50, 60\}$, the neural network has only $\widetilde{D} = 7$ outputs which yield $s_1(k), s_5(k), s_{10}(k), s_{25}(k), s_{40}(k), s_{50}(k), s_{60}(k)$, although the full model has as many as $D = 60$ outputs. In general, the MLP neural network calculates step response coefficients

$$s_{P(p)}(k) = w_{p,0}^2 + \sum_{i=1}^{K} w_{p,i}^2 \varphi(w_{i,0}^1 + w_{i,1}^1 u(k-1) + w_{i,2}^1 y(k)) \tag{7}$$

for $p = 1, \ldots, \widetilde{D}$. The number of hidden nodes is K, weights are denoted by $w_{i,j}^1$, $i = 1, \ldots, K$, $j = 0, 1, 2$, and $w_{i,j}^2$, $i = 0, \ldots, \widetilde{D}$, $j = 0, \ldots, K$, for the first and the second layer, respectively, $\varphi \colon \mathbb{R} \rightarrow \mathbb{R}$ is the nonlinear transfer function (e.g. hyperbolic tangent) used in the hidden layer of the network.

Step response coefficients $s_p \notin P$ are interpolated. For interpolation $\widetilde{D}$ distinct knots

$$P(1) < P(2) < \ldots < P(\widetilde{D} - 1) < P(\widetilde{D}) \tag{8}$$

and corresponding knots values (outputs of the neural model)

$$s_{P(1)}(k) < s_{P(2)}(k) < \ldots < s_{P(\widetilde{D}-1)}(k) < s_{P(\widetilde{D})}(k) \tag{9}$$

are used. Cubic spline interpolation is recommended

$$S(k) = \begin{cases} s_{P(1)}(k) & \text{if} \quad k = [P(1), P(2)] \\ s_{P(2)}(k) & \text{if} \quad k = [P(2), P(3)] \\ \quad \vdots & \\ s_{P(\widetilde{D}-1)}(k) & \text{if} \quad k = [P(\widetilde{D} - 1)), P(\widetilde{D}))] \end{cases} \tag{10}$$

where interpolating polynomials are

$$s_{P(p)}(k) = s_{P(p)}(k) + a_p(k)(k - P(p)) + b_p(k)(k - P(p))^2 + c_p(k)(k - P(p))^3 \tag{11}$$

for all $p = 1, \ldots, \widetilde{D}$, $a_p(k), b_p(k), c_p(k) \in \mathbb{R}$. Cubic splines are smooth in the first derivative and continuous in the second derivative. When compared to classical polynomial interpolation, splines have a few advantages. First, the interpolation error can be made small even when using polynomials of a low degree. Moreover, polynomial interpolation may be not very precise, especially at end points (Runge's phenomenon). Increasing the number of interpolation knots leads to obtaining a polynomial of a high order, which is not a good choice from the numerical point of view. Finally, the cubic spline interpolation problem results in a linear tridiagonal set of equations which can be solved very efficiently [11].

3.3 Optimisation of the Control Policy

Using the step response neural model (6) one obtains predictions [5]

$$\hat{\boldsymbol{y}}(k) = \underbrace{\boldsymbol{G}(k)\triangle\boldsymbol{u}(k)}_{\text{future}} + \underbrace{\boldsymbol{y}(k) + \boldsymbol{G}^{\mathrm{p}}(k)\triangle\boldsymbol{u}^{\mathrm{p}}(k)}_{\text{past}} \tag{12}$$

where $\hat{\boldsymbol{y}}(k) = [\hat{y}(k+1|k)\ldots\hat{y}(k+N|k)]^{\mathrm{T}}$, $\boldsymbol{y}(k) = [y(k)\ldots y(k)]^{\mathrm{T}}$ are vectors of length N, $\triangle\boldsymbol{u}^{\mathrm{p}}(k) = [\triangle u(k-1)\ldots\triangle u(k-(\mathrm{D}-1))]^{\mathrm{T}}$ is a vector of length $\mathrm{D}-1$, matrices

$$\boldsymbol{G}(k) = \begin{bmatrix} s_1(k) & 0 & \ldots & 0 \\ s_2(k) & s_1(k) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ s_N(k) & s_{N-1}(k) & \ldots & s_{N-N_{\mathrm{u}}+1}(k) \end{bmatrix} \tag{13}$$

and

$$\boldsymbol{G}^{\mathrm{p}}(k) = \begin{bmatrix} s_2(k)-s_1(k) & s_3(k)-s_2(k) & \ldots & s_{\mathrm{D}}(k)-s_{\mathrm{D}-1}(k) \\ s_3(k)-s_1(k) & s_4(k)-s_2(k) & \ldots & s_{\mathrm{D}+1}(k)-s_{\mathrm{D}-1}(k) \\ \vdots & \vdots & \ddots & \vdots \\ s_{N+1}(k)-s_1(k) & s_{N+2}(k)-s_2(k) & \ldots & s_{N+\mathrm{D}-1}(k)-s_{\mathrm{D}-1}(k) \end{bmatrix} \tag{14}$$

are of dimensionality $N \times N_{\mathrm{u}}$ and $N \times (D-1)$, respectively.

Using the prediction equation (12), the optimisation problem (3) becomes the following quadratic programming task

$$\begin{aligned} &\min_{\triangle\boldsymbol{u}(k)} \left\{ \left\| \boldsymbol{y}^{\mathrm{ref}}(k) - \boldsymbol{G}(k)\triangle\boldsymbol{u}(k) - \boldsymbol{y}(k) - \boldsymbol{G}^{\mathrm{p}}(k)\triangle\boldsymbol{u}^{\mathrm{p}}(k) \right\|^2 + \left\|\triangle\boldsymbol{u}(k)\right\|^2_{\boldsymbol{\Lambda}} \right\} \\ &\text{subject to} \\ &\quad \boldsymbol{u}^{\min} \leq \boldsymbol{J}\triangle\boldsymbol{u}(k) + \boldsymbol{u}(k-1) \leq \boldsymbol{u}^{\max} \\ &\quad -\triangle\boldsymbol{u}^{\max} \leq \triangle\boldsymbol{u}(k) \leq \triangle\boldsymbol{u}^{\max} \\ &\quad \boldsymbol{y}^{\min} \leq \boldsymbol{G}(k)\triangle\boldsymbol{u}(k) + \boldsymbol{y}(k) + \boldsymbol{G}^{\mathrm{p}}(k)\triangle\boldsymbol{u}^{\mathrm{p}}(k) \leq \boldsymbol{y}^{\max} \end{aligned} \tag{15}$$

where $\boldsymbol{y}^{\mathrm{ref}}(k) = \left[y^{\mathrm{ref}}(k+1|k)\ldots y^{\mathrm{ref}}(k+N|k)\right]^{\mathrm{T}}$, $\boldsymbol{y}^{\min} = \left[y^{\min}\ldots y^{\min}\right]^{\mathrm{T}}$, $\boldsymbol{y}^{\max} = [y^{\max}\ldots y^{\max}]^{\mathrm{T}}$ are vectors of length N, $\boldsymbol{u}^{\min} = \left[u^{\min}\ldots u^{\min}\right]^{\mathrm{T}}$, $\boldsymbol{u}^{\max} = [u^{\max}\ldots u^{\max}]^{\mathrm{T}}$, $\triangle\boldsymbol{u}^{\max} = [\triangle u^{\max}\ldots\triangle u^{\max}]^{\mathrm{T}}$, $\boldsymbol{u}(k-1) = [u(k-1)\ldots u(k-1)]^{\mathrm{T}}$, are vectors of length N_{u}, $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_0, \ldots, \lambda_{N_{\mathrm{u}}-1})$, $\boldsymbol{J}$ is the all ones lower triangular matrix of dimensionality $N_{\mathrm{u}} \times N_{\mathrm{u}}$. To cope with infeasibility problems, output constraints should be softened [7,14].

At each sampling instant k of the neural DMC algorithm the following steps are repeated:

1. For the current operating point of the process use the neural network (7) to calculate $\widetilde{D}$ step response coefficients $s_{P(1)}(k), \ldots, s_{P(\widetilde{D})}(k)$, find remaining coefficients using cubic spline interpolation (10) and (11).
2. Solve the quadratic programming task (15) to find the control policy $\triangle\boldsymbol{u}(k)$.
3. Implement the first element of the obtained vector $u(k) = \triangle u(k|k) + u(k-1)$.
4. Set $k := k+1$, go to step 1.

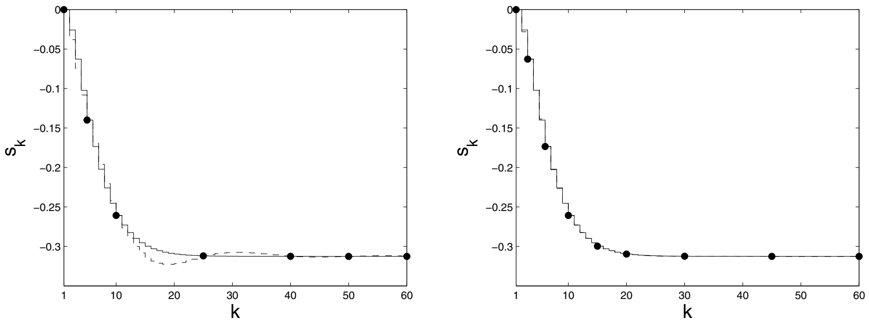


Fig. 1. The scaled step response of the process (*solid line*) vs. the output of the interpolated neural model (*dashed line*) for two sets of interpolation knots: $P_{\mathrm{A}} = \{1, 5, 10, 25, 40, 50, 60\}$ (*left*) and $P_{\mathrm{B}} = \{1, 3, 6, 10, 15, 20, 30, 45, 60\}$ (*right*); interpolation knots are denoted by dots

4 Simulation Results

The process under consideration is a polymerisation reaction taking place in a jacketed continuous stirred tank reactor [8]. The output $NAMW$ (Number Average Molecular Weight) is controlled by manipulating the inlet initiator flow rate F_{I}. Polymerisation is a very important chemical process (production of plastic). Because properties of the reactor are significantly nonlinear, it is frequently used as a benchmark for comparing different control strategies [6,14]. Inefficiency of the classical DMC algorithm based on the linear model is shown in [5].

The fundamental model is used as the real process during simulations. At first, it is simulated open-loop in order to obtain step response coefficients for different operating points. The range of operation is determined by constraints imposed on the manipulated variable: $F_{\mathrm{I}}^{\min} = 0.003$, $F_{\mathrm{I}}^{\max} = 0.06$. The horizon of dynamics is $D = 60$, the sampling time is 1.8 min. Training, validation and test data sets are comprised of 30, 20 and 25 step responses, respectively. The neural model with $K = 3$ hidden nodes is used. The current operating point is defined by $y(k)$, the full model has one input and $D = 60$ outputs which calculate step response coefficients $s_p(k) = f_p(y(k))$, $p = 1, \ldots, D$.

The full step response neural model has 246 parameters (weights). Two interpolated models are also considered. In the first case interpolation knots are $P_{\mathrm{A}} = \{1, 5, 10, 25, 40, 50, 60\}$, in the second case $P_{\mathrm{B}} = \{1, 3, 6, 10, 15, 20, 30, 45, 60\}$. Because the process has a unit delay, i.e. $s_1 = 0$, in the first case the neural network has only $\widetilde{D} - 1 = 6$ outputs (the model has 30 parameters), in the second case $\widetilde{D} - 1 = 8$ outputs (the model has 38 parameters). As in the full model $K = 3$ hidden nodes are used in the hidden layer. Fig. 1 depicts the scaled step response of the process in the nominal operating point ($NAMW = 20000$) vs. the output of the model for two sets of interpolation knots (P_{A} and P_{B}). In the first case the model is not very precise, increasing the number of knots improves accuracy.

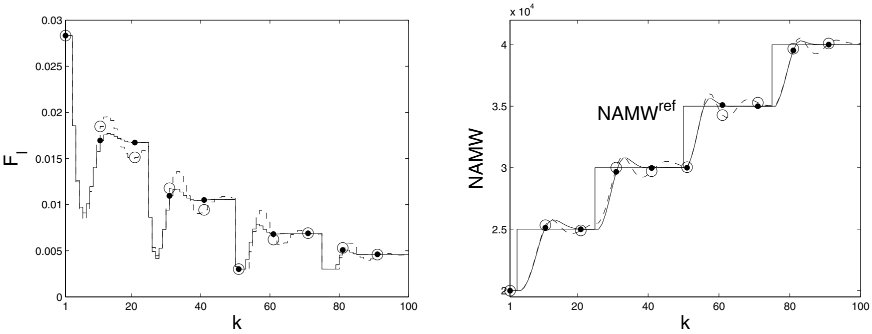


Fig. 2. Simulation results of the nonlinear DMC algorithm based on: the full step response neural model (*solid line with dots*) and the interpolated model with interpolation knots $P_{\mathrm{A}} = \{1, 5, 10, 25, 40, 50, 60\}$ (*dashed line with circles*)

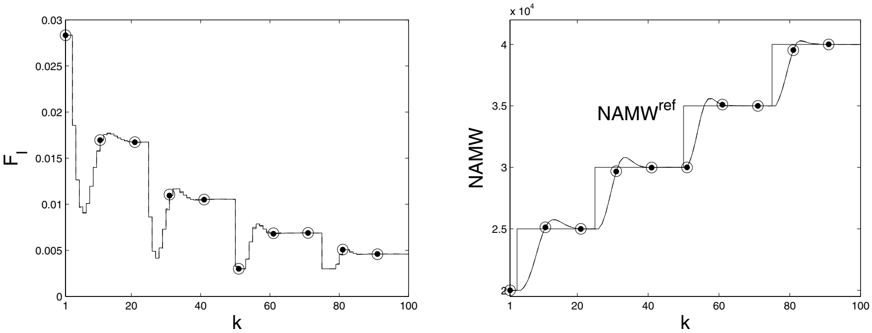


Fig. 3. Simulation results of the nonlinear DMC algorithm based on: the full step response neural model (*solid line with dots*) and the interpolated model with interpolation knots $P_{\mathrm{B}} = \{1, 3, 6, 10, 15, 20, 30, 45, 60\}$ (*dashed line with circles*); trajectories obtained in both algorithms are practically the same

In the neural DMC algorithm three different models are used: the full step response neural model and interpolated models with knots P_{A} and P_{B}. Tuning parameters are: $N = 10$, $N_{\mathrm{u}} = 3$, $D = 60$, $\lambda_p = 0.2$. Simulation results are depicted in Fig. 2 and Fig. 3. The algorithm based on the full step response neural model is stable and precise. The smaller set of interpolation knots gives some modelling errors, hence, when it is used for control, accuracy is not sufficient. Increasing the number of interpolation knots gives very good control, in fact trajectories obtained in nonlinear DMC algorithms based on the full step response neural model and interpolated model with interpolation knots P_{B} are practically the same as it is shown in Fig. 3.

5 Conclusions

This paper describes a nonlinear DMC algorithm based on the interpolated step response neural model. Because step response models typically need many coefficients, interpolation makes it possible to reduce model complexity. In this approach only selected step response coefficients are actually calculated by the neural network whereas remaining ones are interpolated by cubic splines. The presented approach is particularly interesting in multivariable processes control. In such cases full step response neural models may have hundreds of weights.

Acknowledgement. The work presented in this paper was supported by Polish national budget funds for science for years 2009-2011.

References

1. Blevins, T.L., Mcmillan, G.K., Wojsznis, M.W.: Advanced control unleashed. ISA (2003)
2. Cutler, C.R., Ramaker, B.L.: Dynamic matrix control – a computer control algorithm. In: Proceedings of the Joint Automatic Control Conference, San Francisco, USA (1979)
3. Haykin, S.: Neural networks – a comprehensive foundation. Prentice Hall, Englewood Cliffs (1999)
4. Henson, M.A.: Nonlinear model predictive control: current status and future directions. Computers and Chemical Engineering 23, 187–202 (1998)
5. Ławryńczuk, M.: Neural dynamic matrix control algorithm with disturbance compensation. In: IEA/AIE 2010. LNCS (LNAI). Springer, Heidelberg (2010)
6. Ławryńczuk, M.: A family of model predictive control algorithms with artificial neural networks. International Journal of Applied Mathematics and Computer Science 17, 217–232 (2007)
7. Maciejowski, J.M.: Predictive control with constraints. Prentice Hall, Englewood Cliffs (2002)
8. Maner, B.R., Doyle, F.J., Ogunnaike, B.A., Pearson, R.K.: Nonlinear model predictive control of a simulated multivariable polymerization reactor using second-order Volterra models. Automatica 32, 1285–1301 (1996)
9. Marusak, P.: Advantages of an easy to design fuzzy predictive algorithm in control systems of nonlinear chemical reactors. Applied Soft Computing 9, 1111–1125 (2009)
10. Nørgaard, M., Ravn, O., Poulsen, N.K., Hansen, L.K.: Neural networks for modelling and control of dynamic systems. Springer, London (2000)
11. Press, W.H., Teukolsky, S.A., Vettering, W.T., Flannery, B.P.: Numerical recipes in C: the art of scientific computing. Cambridge University Press, Cambridge (1992)
12. Qin, S.J., Badgwell, T.A.: A survey of industrial model predictive control technology. Control Engineering Practice 11, 733–764 (2003)
13. Rossiter, J.A.: Model-based predictive control. CRC Press, Boca Raton (2003)
14. Tatjewski, P.: Advanced control of industrial processes, structures and algorithms. Springer, London (2007)

Approximate Neural Economic Set-Point Optimisation for Control Systems

Maciej Ławryńczuk and Piotr Tatjewski

Institute of Control and Computation Engineering, Warsaw University of Technology
ul. Nowowiejska 15/19, 00-665 Warsaw, Poland, tel. +48 22 234-76-73
M.Lawrynczuk@ia.pw.edu.pl, P.Tatjewski@ia.pw.edu.pl

Abstract. This paper describes a neural approach to economic set-point optimisation which cooperates with Model Predictive Control (MPC) algorithms. Because of high computational complexity, nonlinear economic optimisation cannot be repeated frequently on-line. Alternatively, an additional steady-state target optimisation based on a linear or a linearised model and repeated as often as MPC is usually used. Unfortunately, in some cases such an approach results in constraint violation and numerical problems. The approximate neural set-point optimiser replaces the whole nonlinear economic set-point optimisation layer.

Keywords: Process control, set-point optimisation, Model Predictive Control, neural networks, optimisation.

1 Introduction

Model Predictive Control (MPC) algorithms have been successfully used for years in numerous advanced industrial applications [9,10,13]. It is mainly because they have a unique ability to take into account constraints imposed on process inputs (manipulated variables) and outputs (controlled variables) which determine quality, economic efficiency and safety. Moreover, MPC techniques are very efficient in multivariable process control. Usually, MPC algorithms use linear or linearised dynamic models of the process. As a result, MPC optimisation is in fact an easy to solve quadratic programming problem. Inaccuracies resulted from an unavoidable mismatch between the model used in MPC and properties of the process are naturally compensated by the negative feedback loop.

To maximise economic gains MPC algorithms cooperate with economic set-point optimisation, which calculates on-line set-points for MPC [1,2,3,12,13]. Although in the simplest case set-point optimisation uses a constant linear steady-state model derived from the dynamic model used for MPC [5,7,12,13], in general a comprehensive nonlinear steady-state model is used. In consequence, for set-point calculation a nonlinear optimisation problem must be solved in open-loop (no feedback). Preferably, set-point optimisation should be repeated on-line as often as MPC is activated. Because of high computational complexity, it is usually not possible. Low frequency of economic optimisation can be economically inefficient when disturbances (e.g. flow rates, properties of feed and energy streams) vary significantly and fast [13].

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 305–312, 2010.
© Springer-Verlag Berlin Heidelberg 2010

To reduce the computational burden of nonlinear set-point optimisation a linearised on-line steady-state model can be used. In such a case the set-point optimisation task becomes a quadratic programming problem which can be solved as often as MPC is activated [7,8,12,13]. Unfortunately, in some cases linearisation-based approaches result in constraint violation and numerical problems (inaccuracies are not compensated by the negative feedback loop as it is done in MPC). They can be eliminated in set-point optimisation based on a piecewise linear approximation of the nonlinear steady-state model [6].

This paper presents a computationally efficient alternative to the classical set-point optimisation in which a nonlinear model of the process is used – an approximate neural set-point optimiser. It replaces the whole nonlinear set-point optimisation layer and it can be activated as often as MPC. The neural set-point optimiser is trained off-line, no on-line optimisation is necessary. As an example process a multivariable chemical reactor is studied.

2 Classical Multilayer Control System Structure

The standard multilayer control system structure is depicted in Fig. 1. The objective of economic optimisation (named local steady-state optimisation [13]) is to maximise the production profit J_{E} and to satisfy constraints, which determine safety and quality of production. The economic optimisation problem is

$$\begin{aligned}&\min_{u^{\mathrm{ss}}}\left\{J_{\mathrm{E}}=c_{\mathrm{u}}^{\mathrm{T}}u^{\mathrm{ss}}-c_{\mathrm{y}}^{\mathrm{T}}y^{\mathrm{ss}}\right\}\\ &\text{subject to}\\ &\quad u^{\min}\leq u^{\mathrm{ss}}\leq u^{\max}\\ &\quad y^{\min}\leq y^{\mathrm{ss}}\leq y^{\max}\\ &\quad y^{\mathrm{ss}}=f^{\mathrm{ss}}(u^{\mathrm{ss}},h^{\mathrm{ss}})\end{aligned}\tag{1}$$

where $u=\left[u_1\ldots u_{n_{\mathrm{u}}}\right]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathrm{u}}}$ are inputs of the process (manipulated variables), $y=\left[y_1\ldots y_{n_{\mathrm{y}}}\right]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathrm{y}}}$ are outputs (controlled variables), $h=\left[h_1\ldots h_{n_{\mathrm{h}}}\right]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathrm{h}}}$ are measured or estimated disturbances, the superscript 'ss' refers to the steady-state, vectors $c_{\mathrm{u}}=\left[c_{\mathrm{u},1}\ldots c_{\mathrm{u},n_{\mathrm{u}}}\right]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathrm{u}}}$, $c_{\mathrm{y}}=\left[c_{\mathrm{y},1}\ldots c_{\mathrm{y},n_{\mathrm{y}}}\right]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathrm{y}}}$ represent economic prices, constraints are defined by vectors

$$u^{\min}=\left[u_1^{\min}\ldots u_{n_{\mathrm{u}}}^{\min}\right]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathrm{u}}},\qquad y^{\min}=\left[y_1^{\min}\ldots y_{n_{\mathrm{y}}}^{\min}\right]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathrm{y}}}\tag{2}$$

$$u^{\max}=\left[u_1^{\max}\ldots u_{n_{\mathrm{u}}}^{\max}\right]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathrm{u}}},\qquad y^{\max}=\left[y_1^{\max}\ldots y_{n_{\mathrm{y}}}^{\max}\right]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathrm{y}}}\tag{3}$$

A comprehensive nonlinear steady-state model of the process is denoted by the function $f^{\mathrm{ss}}\colon\mathbb{R}^{n_{\mathrm{u}}+n_{\mathrm{h}}}\longrightarrow\mathbb{R}^{n_{\mathrm{y}}}$.

In MPC algorithms [9,13] at each sampling instant k future control increments are calculated in such a way that differences between the economically optimal set-point $y_{\mathrm{lsso}}^{\mathrm{ss}}(k)$ (corresponding to $u_{\mathrm{lsso}}^{\mathrm{ss}}(k)$) and output predictions over some time horizon are minimised on-line subject to input and output constraints.

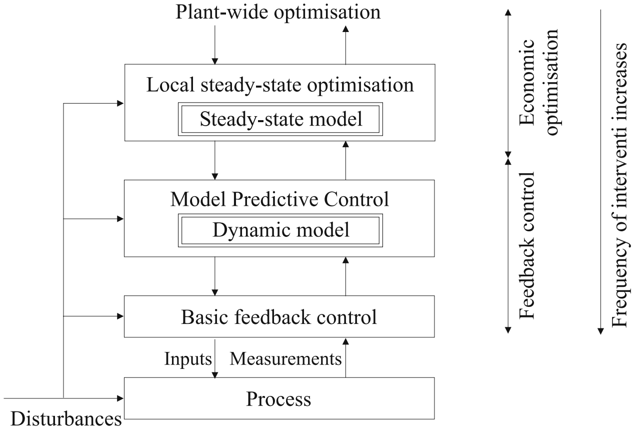


Fig. 1. The classical multilayer control system structure

3 Approximate Neural Set-Point Optimisation

The idea of the described approach is eliminate the necessity of solving on-line the nonlinear set-point economic optimisation problem (1) carried out at the local steady-state optimisation layer shown in Fig. 1. On-line set-point optimisation is replaced by an approximate neural optimiser, the objective of which is to calculate economically optimal set-points. In other words, for current operating conditions of the process (determined by measured or estimated values of disturbances) and constraints, the neural optimiser calculates the solution to the set-point optimisation problem in an approximate way, the necessity of numerical optimisation is eliminated. Thanks to it, the computational burden of the neural optimiser is very low, it can be activated as often as MPC.

The solution to the nonlinear set-point economic optimisation problem (1) is a function of disturbances and constraints imposed on inputs and outputs of the process. Hence, the neural set-point optimiser approximates the relation

$$y^{\mathrm{ss}} = g^{\mathrm{ss}}(h^{\mathrm{ss}}, u^{\min}, u^{\max}, y^{\min}, y^{\max}) \tag{4}$$

where $g^{\mathrm{ss}} \colon \mathbb{R}^{n_{\mathrm{h}}+2n_{\mathrm{u}}+2n_{\mathrm{y}}} \rightarrow \mathbb{R}^{n_{\mathrm{y}}}$. In this work relations (4) are realised by n_{y} MultiLayer Perceptron (MLP) feedforward neural networks [4]. Each network has $n_{\mathrm{h}} + 2n_{\mathrm{u}} + 2n_{\mathrm{y}}$ input nodes, one hidden layer and a linear output.

The neural set-point optimiser is trained off-line, no on-line set-point optimisation is necessary. At first, the set-point economic optimisation problem (1) is solved off-line for a set of different values of disturbances and constraints. Next, obtained data sets are used for neural network training, validation and testing. Finally, the chosen neural optimiser is used on-line to calculate economically optimal set-points for MPC.

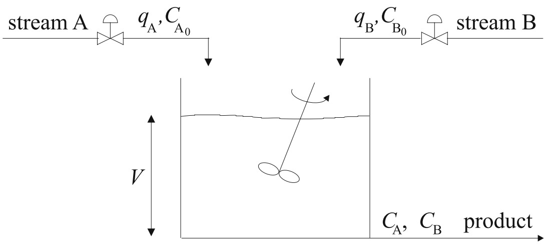


Fig. 2. The chemical reactor

4 Simulation Results

The considered technological process is an isothermal, constant-volume, continuous, stirred-tank reactor shown in Fig. 2 [11]. The reactor has two feed streams (A and B). C_{A} and C_{B} are concentrations of reactants A and B in the product stream, respectively, C_{A_0} is concentration of the reactant A in the feed stream A and C_{B_0} is concentration of the reactant B in the feed stream B, q_{A} and q_{B} are volumetric flow rates of feed streams, respectively and V is the volume of the reactor. Nominal concentrations are: $C_{\mathrm{A}_0} = 1 \, \frac{\mathrm{kmol}}{\mathrm{m}^3}$, $C_{\mathrm{B}_0} = 1.2 \, \frac{\mathrm{kmol}}{\mathrm{m}^3}$.

The process has two input (manipulated) variables: $v_1 = \frac{q_{\mathrm{A}}}{V}$, $v_2 = \frac{q_{\mathrm{B}}}{V}$ and two output (controlled) variables: C_{A} and C_{B} (i.e. $u = [v_1 \ v_2]^{\mathrm{T}}$, $y = [C_{\mathrm{A}} \ C_{\mathrm{B}}]^{\mathrm{T}}$). Manipulated variables are constrained

$$v_1^{\min} \leq v_1 \leq v_1^{\max}, \qquad v_2^{\min} \leq v_2 \leq v_2^{\max} \tag{5}$$

where $v_1^{\min} = 0 \, \frac{1}{\mathrm{s}}$, $v_1^{\max} = 1.9 \cdot 10^{-3} \, \frac{1}{\mathrm{s}}$, $v_2^{\min} = 0 \, \frac{1}{\mathrm{s}}$, $v_2^{\max} = 6 \cdot 10^{-4} \, \frac{1}{\mathrm{s}}$. Concentrations C_{A_0} and C_{B_0} of reactants A and B in feed streams may change. Hence, they are treated as disturbances. It is assumed that values of disturbances may change in time by ±40% from their nominal values

$$0.6 \, \frac{\mathrm{kmol}}{\mathrm{m}^3} \leq C_{\mathrm{A}_0} \leq 1.4 \, \frac{\mathrm{kmol}}{\mathrm{m}^3}, \qquad 0.8 \, \frac{\mathrm{kmol}}{\mathrm{m}^3} \leq C_{\mathrm{B}_0} \leq 1.6 \, \frac{\mathrm{kmol}}{\mathrm{m}^3} \tag{6}$$

To maximise the production rate, for set-point optimisation the economic performance function (1) with $c_{\mathrm{u}} = [-1 \ -1]^{\mathrm{T}}$ and $c_{\mathrm{y}} = [0 \ 0]^{\mathrm{T}}$ is used, i.e.

$$J_{\mathrm{E}} = -v_1^{\mathrm{ss}} - v_2^{\mathrm{ss}} \tag{7}$$

It means that economic optimisation maximises the amount of the substance flowing through the reactor and, in consequence, the amount of production. A composition constraint is also imposed

$$C_{\mathrm{A}}^{\min} \leq C_{\mathrm{A}}^{\mathrm{ss}} \tag{8}$$

It is assumed that the current value of this constraint may be selected by the operator of the process from the following range

$$0.45 \, \frac{\mathrm{kmol}}{\mathrm{m}^3} \leq C_{\mathrm{A}}^{\min} \leq 0.55 \, \frac{\mathrm{kmol}}{\mathrm{m}^3} \tag{9}$$

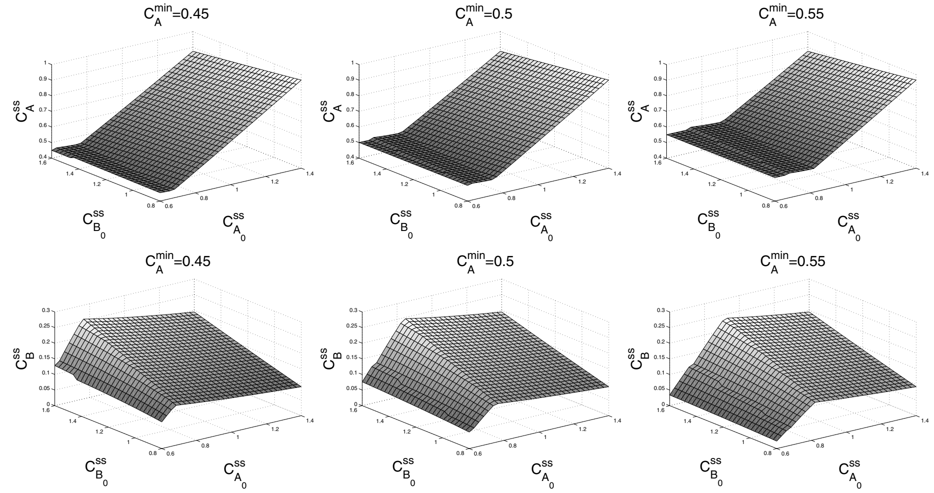


Fig. 3. Optimal set-points for MPC ($C_{\mathrm{A}}^{\mathrm{ss}}$, $C_{\mathrm{B}}^{\mathrm{ss}}$) as function of disturbances ($C_{\mathrm{A}_0}^{\mathrm{ss}}, C_{\mathrm{B}_0}^{\mathrm{ss}}$) for three different values of the output constraint $C_{\mathrm{A}}^{\mathrm{min}} = 0.45$, $C_{\mathrm{A}}^{\mathrm{min}} = 0.5$, $C_{\mathrm{A}}^{\mathrm{min}} = 0.55$

Taking into account (7), (5) and (8), for the chemical reactor the economic set-point optimisation problem (1) becomes

$$\begin{aligned}
&\min_{v_1^{\mathrm{ss}},\ v_2^{\mathrm{ss}}} \{J_{\mathrm{E}} = -v_1^{\mathrm{ss}} - v_2^{\mathrm{ss}}\}\\
&\text{subject to}\\
&\quad v_1^{\mathrm{min}} \leq v_1^{\mathrm{ss}} \leq v_1^{\mathrm{max}}\\
&\quad v_2^{\mathrm{min}} \leq v_2^{\mathrm{ss}} \leq v_2^{\mathrm{max}}\\
&\quad C_{\mathrm{A}}^{\mathrm{min}} \leq C_{\mathrm{A}}^{\mathrm{ss}}\\
&\quad C_{\mathrm{A}}^{\mathrm{ss}} = f_{\mathrm{A}}^{\mathrm{ss}}(v_1^{\mathrm{ss}}, v_2^{\mathrm{ss}})
\end{aligned} \tag{10}$$

Having obtained optimal values of process inputs (v_1^{ss}, v_2^{ss}), optimal set-points for MPC ($C_{\mathrm{A}}^{\mathrm{ss}}$, $C_{\mathrm{B}}^{\mathrm{ss}}$) are calculated using steady-state characteristics $C_{\mathrm{A}}^{\mathrm{ss}} = f_{\mathrm{A}}^{\mathrm{ss}}(v_1^{\mathrm{ss}}, v_2^{\mathrm{ss}})$ and $C_{\mathrm{B}}^{\mathrm{ss}} = f_{\mathrm{B}}^{\mathrm{ss}}(v_1^{\mathrm{ss}}, v_2^{\mathrm{ss}})$.

The output constraint $C_{\mathrm{A}}^{\mathrm{min}} \leq C_{\mathrm{A}}^{\mathrm{ss}}$, due to nonlinearity of the steady-state characteristics, is nonlinear. Hence, the set-point optimisation problem (10) is a nonlinear task. In the classical multilayer structure, the set-point optimisation task must be solved on-line for current values of disturbances C_{A_0}, C_{B_0} and for the current value $C_{\mathrm{A}}^{\mathrm{min}}$. The neural set-point optimiser approximates the relation (4), for the considered process it calculates optimal set-points as functions

$$C_{\mathrm{A}}^{\mathrm{ss}} = g_{\mathrm{A}}^{\mathrm{ss}}(C_{\mathrm{A}_0}^{\mathrm{ss}}, C_{\mathrm{B}_0}^{\mathrm{ss}}, C_{\mathrm{A}}^{\mathrm{min}}), \qquad C_{\mathrm{B}}^{\mathrm{ss}} = g_{\mathrm{B}}^{\mathrm{ss}}(C_{\mathrm{A}_0}^{\mathrm{ss}}, C_{\mathrm{B}_0}^{\mathrm{ss}}, C_{\mathrm{A}}^{\mathrm{min}}) \tag{11}$$

Fig. 3 depicts optimal set-points for MPC ($C_{\mathrm{A}}^{\mathrm{ss}}$, $C_{\mathrm{B}}^{\mathrm{ss}}$) as function of disturbances ($C_{\mathrm{A}_0}^{\mathrm{ss}}, C_{\mathrm{B}_0}^{\mathrm{ss}}$) for three different values of the output constraint $C_{\mathrm{A}}^{\mathrm{min}}$.

The approximate neural set-point optimiser is comprised of two neural networks who realise functions $g_{\mathrm{A}}^{\mathrm{ss}}$, $g_{\mathrm{B}}^{\mathrm{ss}}$. The set-point optimisation task (10) is solved off-line for different values of disturbances (6) and the output constraint (8) selected from the range (9). Obtained data set is divided into three sets (training, validation and test data sets), each set has 8000 samples. As a compromise between accuracy and complexity neural networks with 5 hidden nodes are used.

The MPC algorithm is of the GPC type [13]. The sampling period is 10 seconds. The same constraints (5), (8) used in set-point optimisation are used in MPC. As the simulated process the fundamental model [11] is used.

Three different structures are compared

a) The structure with set-point economic optimisation based on a linearised on-line the comprehensive nonlinear steady-state model of the process [7,8,13]. In such a case set-point optimisation is a linear-programming task.
b) The structure with set-point economic optimisation based on the comprehensive nonlinear steady-state model of the process. In such a case set-point optimisation is a nonlinear optimisation task.
c) The structure with the approximate neural set-point optimiser which replaces the whole set-point optimisation layer. No set-point optimisation task is solved on-line. Set-points are calculated easily by neural networks who realise functions $g_{\mathrm{A}}^{\mathrm{ss}}$, $g_{\mathrm{B}}^{\mathrm{ss}}$ in (11).

In all three structures set-points are calculated as frequently as the MPC algorithm is activated. The scenario of disturbance changes is

$$C_{\mathrm{A}_0}(k) = 1 - 0.4(\sin(0.015k)), \qquad C_{\mathrm{B}_0}(k) = 1.2 - 0.4(\sin(0.015k)) \tag{12}$$

The simulation horizon is 150 discrete time steps. For the first part of simulations ($k = 1, \ldots, 99$), the output constraint $C_{\mathrm{A}}^{\min} = 0.5$. For the second part of simulations ($k = 100, \ldots, 150$), $C_{\mathrm{A}}^{\min} = 0.54$.

Fig. 4 shows simulation results. Unfortunately, the linearisation-based approach is numerically unreliable. In the vicinity of the sampling instant $k = 50$ the calculated set-point violates the output constraint $C_{\mathrm{A}}^{\min} \leq C_{\mathrm{A}}^{\mathrm{ss}}$. From the sampling instant $k = 50$ trajectories (in particular $C_{\mathrm{A}}^{\mathrm{ss}}$) are different from optimal ones (the second structure). For the sampling instant $k = 100$ the output constraint is violated and till the end of simulations the whole optimisation problem is infeasible (the set of possible solutions determined by constraints is empty, set-point optimisation is unable to find the solution). It is because the steady-state model is nonlinear and inaccuracies of the linearised model are big.

Trajectories calculated by the approximate neural set-point optimiser are very close to those obtained when for set-point optimisation the full nonlinear model is used without any simplifications. Having completed simulations, the economic performance index $J_{\mathrm{E}}^{\mathrm{sim}} = \sum_{k=1}^{150} (-v_1(k) - v_2(k))$ corresponding to the cost-function (7) which is actually minimised during on-line set-point optimisation is calculated. Table 1 compares values of $J_{\mathrm{E}}^{\mathrm{sim}}$ and the computational burden obtained in the classical structure with nonlinear set-point optimisation and by the approximate neural set-point optimiser. The approximate neural approach gives economic results only slightly worse in comparison with the classical structure an important disadvantage of which is high computational load (nonlinear set-point optimisation repeated as frequently as MPC executes).

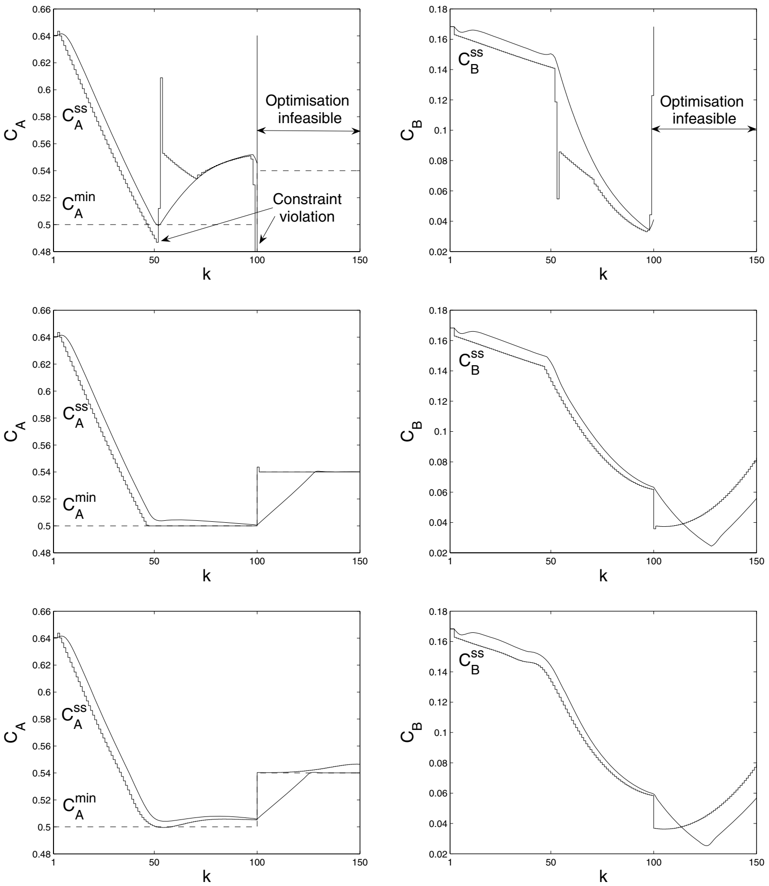


Fig. 4. Simulation results: set-point optimisation based on a linearised model (*top panels*), set-point optimisation based on a nonlinear model (*middle panels*), approximate neural set-point optimisation (*bottom panels*)

Table 1. The economic performance index $J_{\mathrm{E}}^{\mathrm{sim}}$ and the computational burden CB (in MFLOPS) in the classical structure with nonlinear set-point optimisation and in the structure with approximate neural set-point optimisation

Structure	$J_{\mathrm{E}}^{\mathrm{sim}}$	CB
Nonlinear set-point optimisation	-2.0243	2.8138
Approximate neural set-point optimisation	-2.0192	0.0309

5 Conclusions

For the considered chemical reactor the linearisation-based approach to set-point economic optimisation gives numerically wrong results whereas set-point optimisation based on a nonlinear model is computationally demanding. The neural set-point optimiser gives system trajectories and economic efficiency similar to those obtained when nonlinear optimisation is used on-line. The neural optimiser is computationally very efficient, for current values of disturbances and constraints, neural networks calculate approximations of optimal set-points. As a result, the necessity of on-line nonlinear set-point optimisation is eliminated.

Acknowledgement. The work presented in this paper was supported by Polish national budget funds for science for years 2009-2011.

References

1. Blevins, T.L., Mcmillan, G.K., Wojsznis, M.W.: Advanced control unleashed. ISA (2003)
2. Brdys, M., Tatjewski, P.: Iterative algorithms for multilayer optimizing control. Imperial College Press, London (2005)
3. Findeisen, W.M., Bailey, F.N., Brdyś, M., Malinowski, K., Tatjewski, P., Woźniak, A.: Control and coordination in hierarchical systems. J. Wiley and Sons, Chichester (1980)
4. Haykin, S.: Neural networks – a comprehensive foundation. Prentice-Hall, Englewood Cliffs (1999)
5. Kassmann, D.E., Badgwell, T.A., Hawkins, R.B.: Robust steady-state target calculation for model predictive control. AIChE Journal 46, 1007–1024 (2000)
6. Ławryńczuk, M., Marusak, P., Tatjewski, P.: Piecewise linear steady-state target optimization for control systems with MPC: a case study. In: The 17th IFAC World Congress, Seoul, South Korea, DVD-ROM, paper 3204, pp. 13169–13174 (2008)
7. Ławryńczuk, M., Marusak, P., Tatjewski, P.: Cooperation of model predictive control with steady-state economic optimisation. Control and Cybernetics 37, 133–158 (2008)
8. Ławryńczuk, M.: Neural models in computationally efficient predictive control cooperating with economic optimisation. In: de Sá, J.M., Alexandre, L.A., Duch, W., Mandic, D.P. (eds.) ICANN 2007. LNCS, vol. 4669, pp. 650–659. Springer, Heidelberg (2007)
9. Maciejowski, J.M.: Predictive control with constraints. Prentice-Hall, Harlow (2002)
10. Qin, S.J., Badgwell, T.A.: A survey of industrial model predictive control technology. Control Engineering Practice 11, 733–764 (2003)
11. Soroush, M., Valluri, S., Nehranbod, N.: Nonlinear control of input-constrained systems. Computers and Chemical Engineering 30, 158–181 (2005)
12. Tatjewski, P.: Advanced control and on-line process optimization in multilayer structures. Annual Reviews in Control 32, 71–85 (2008)
13. Tatjewski, P.: Advanced control of industrial processes, Structures and algorithms. Springer, London (2007)

Injecting Service-Orientation into Multi-Agent Systems in Industrial Automation

J. Marco Mendes[1], Francisco Restivo[1],
Paulo Leitão[2], and Armando W. Colombo[3]

[1] Faculty of Engineering - University of Porto,
Rua Dr. Roberto Frias s/n, 4200-465 Porto, Portugal
{marco.mendes,fjr}@fe.up.pt

[2] Polytechnic Institute of Bragança, Quinta Sta Apolónia,
Apartado 134, 5301-857 Bragança, Portugal
pleitao@ipb.pt

[3] Schneider Electric Automation GmbH,
Steinheimer Str. 117, D-63500 Seligenstadt, Germany
armando.colombo@de.schneider-electric.com

Abstract. Service-oriented architecture and multi-agent systems are used in the research of novel control systems for industrial automation. This work presents the advance of service-orientation into multi-agent system in industrial automation. An overview of the concept of service-oriented agent is done and also the enhancements that services can bring to multi-agent systems. The documented topics of this work are based on the experiments and projects of the authors in the domain of distributed and component based automation systems. The outcome shows that there are many benefits, namely the aspects of knowledge representation, communication, plug & play, interactions, beside others. These contribute to the vision of future automation systems, where flexibility and customization are strong arguments.

Keywords: multi-agent systems, service-oriented architecture, service-oriented computing, industrial automation.

1 Introduction

Today's control applications of industrial automation are usually developed in the form of large monolithic software packages that are difficult to be maintained, modified, and extended. Therefore a movement from distributed systems controlled by users to automatic, autonomous and self-configuring distributed systems is noticeable [1]. Experiments are done with distributable and component-based approaches, such as multi-agent systems (MAS) and service-oriented architectures (SOA). These systems are targeting the flexibility and will be created using basic process modules — hardware and software — that will be rearranged quickly and reliably [2]. This initiative is also reflected in the concept of Collaborative Automation [3] in the sense of autonomous, reusable, loosely-coupled, distributed and collaborative components.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 313–320, 2010.
© Springer-Verlag Berlin Heidelberg 2010

MAS and SOA have been a major impulse to renew the older and centralistic PLC (Programmable Logic Controller) and IEC 61131-based systems, but also to challenge the industrial standard of the IEC 61499 function blocks. The "fusion" between several trends has been always a point of research, wherever it makes more or less sense. MAS and SOA are one combination that has been seen as natural and beneficial. Several works were done (and continue to be realized) by blending both of them. One perspective suggests interoperability mechanisms between the agent system and service system, using specific gateways/proxies to interface both worlds (see the work of [4]). In this case, MAS is working aside from the SOA and both are not directly intersected. Another possibility is the encapsulation of agents into services and thus communicating in a "service-way" (for example, the "Agents as Web services" in [5]). The last one that is found in research publications is more close to the concept pretended in this paper, namely service-oriented agents. These do not only share services as their major form of communication, but also complement their own goals with external provided services (see [6] and [7]).

In the research works by the authors in the application of software components in industrial automation (independently if they were classified as agents or not), a common feature was that they have a domain of autonomy regarding some automation component, machinery, product or a set of several composed entities. This makes them similar to agents and, since they require coordination aspects, it is possible to talk about a multi-agent system. Recently, the involvement in the EU research project SOCRADES (Service-oriented cross-layer infrastructure for distributed smart embedded systems), smart software components were embedded into automation devices and particularly enriched with service-orientation (not only using web service as communication form, but also other engineering aspects, such as representation of services by resource availability, composition, orchestration, etc.). In this case it makes sense referring to a community of service-oriented agents, in other words, service-oriented multi-agent systems (SoMAS) [7].

So, what is a service-oriented agent anyway? By consulting recent bibliography, there is no clear reference. Therefore, in this work a service-oriented agent is understood as an agent that behaves accordantly to the concept of service-orientation. In other words, an agent represents its conditions and resources that it want to share in form of services, and, when it requires external support, it will try to find and request services available in the system (possibly also exposed by other agents). Agents are not only problem solvers in the limited sense, but also are considered as mediators or responsible entities of a particular system.

Fig. 1 shows a representation of a service-oriented agent located in his environment. For the considered domain of this paper, Fig. 1 will be explained recurring to the example of industrial automation. The environment is an industrial automation system made of control devices, machinery, conveyors and other equipments and resources. A service-oriented agent is responsible of part of this environment (domain of autonomy), so easily it could be defined for example, as a mediator of a conveyor segment. Therefore it has the ability to read the

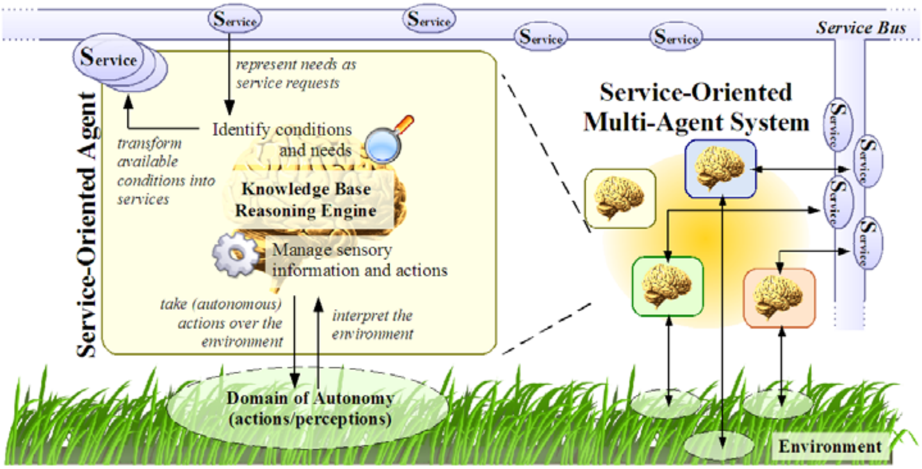

Service
Service
Service
Service
Service
Service Bus
Service-Oriented Agent
represent needs as service requests
transform available conditions into services
Identify conditions and needs
Knowledge Base
Reasoning Engine
Manage sensory information and actions
take (autonomous) actions over the environment
interpret the environment
Domain of Autonomy (actions/perceptions)
Service-Oriented Multi-Agent System
Environment

Fig. 1. Representation of a service-oriented agent and its environment

sensors and control the actuators of the conveyor, to make it possible to transport pallets from its input to its output. The previous sentence can be defined as the internal objective of the agent, but it has also to respect external/global objectives of the system. The objective and available condition can be offered as a service to the outside (service: transport pallets), so that possible another entity (e.g. a pallet agent) could request it ("Please transport me from point A to point B"). However to complete the service and also to respect global objectives, the conveyor must request an availability service from the next transport unit or workstation connected to its output. This can be seen as the form of collaboration and automatic rearrangement of service in this system.

From several research projects and industrial experiments of component-based approaches (such as the SOCRADES project, www.socrades.eu) some intersections can be made. A particular appointment is that agent systems have some common features with recently developed service systems in industrial automation. What is still missing and needs to be answered is: when having multi-agent system, what are the benefits of injecting it with service-orientation/technology? The answer cannot be singularly limited to the obvious "communication", but must be extended into a complete philosophy around services: from an orientation principle for agents to a collaborative response (as explained previously), besides the technological aspects. The service-oriented behavior of these agents is not the only feature that can be introduced by adapting a real SOA to a multi-agent system. From the authors' experience and observations in the development of distributed automation systems founded on autonomous entities, other topics can be discussed that involves technological and conceptual elements that compose SOA. Therefore, the main kernel of this paper will be the topics of service-orientation that were identified on the developed projects and that are seen as a major point of discussion in the integration into multi-agent systems.

2 Enhancing Agents with Service-Orientation

In [7] was identified some of the topics of engineering that could be addressed in SoMAS, but without the focus on introducing already known SOA technologies in MAS systems. This section will discuss these features and retrieve the benefits of the enrichment of MAS with SOA. However, it is not intended to highlight the characteristic that (intelligent) agents should normally manifest (e.g. autonomy, intelligence, social ability, pro-activity), but more the introductions from the SOA perspective.

2.1 Technology and Standards

Since both concepts of MAS and SOA define a set of properties that should be observed on implemented systems, there are obviously requirements in the type of technology standards and protocols that such systems should be based on. Most SOA-based systems are implemented using web services, which are made of a set of XML and specific protocols. MAS are usually compliant to the syntax defined by FIPA (Foundation for Intelligent Physical Agents). It is possible to make an analogy between similar protocols used in MAS and SOA. For example, in FIPA there are agent and service directory services, providing the ability to register and search for agents/services. For web service, Universal Description, Discovery and Integration (UDDI) is used to provide a similar feature. This relationship is compared in more detail in [8].

In the home and industrial automation domain, the main technological specification that is used in the research of service-based devices is the Device Profile for Web Services (DPWS) [9]. The specification permits the definition of web services for devices considering the peer-to-peer direct communication between them. DPWS allows sending secure messages to and from web services, dynamically discovering a web service, describing a web service, subscribing to, and receiving events from a web service [1].

The benefits of using these web service technologies integrated in MAS can be understood as the uniform and standard way of communication. Agents would profiteer from the available SOA technologies and implementations, not only for means of communication, but from the numerous web service and adjacent protocols covering most of the requirements.

2.2 Knowledge Representation

The use of an XML-based description of resources, actions, services and everything else in the system can facilitate the representation of knowledge for internal use of an agent and also for interchange with other entities. Web Services Description Language (WSDL) can be seen as one of the pillars of web service by describing them in XML format that can be easily understood by humans and non-humans (software applications). In the case of web service descriptions, automated matching and service selection based on WSDL illustrate that the semantic annotation of web services can provide a sound basis for intelligent agents able to automate mediation, choreography, and discovery processes between web services [10]. Especially in automation, when the devices/machines to be controlled are significantly heterogeneous, a momentous amount of customized control code needs to be produced, and the glue code portion can become very complex [11]. This should be handled by rich descriptions and automatic reasoning systems over these control codes. It is clear that the potential of the wide available description forms are prone for the knowledge base and reasoning engines of agents to make their choices.

2.3 Dynamic Deployment, Dynamic Discovery and Plug and Play

Several useful features that were demonstrated during the SOCRADES project were the ability of the automation entities being dynamically configured and discovered. These automation entities with agent-like properties were embedded into industrial control devices that are mediators of one or more equipments and use DPWS for service-based communication with other entities in the system via the network.

One of the improvements that are expected from the service-oriented approach is a greater flexibility in reconfiguring and upgrading services. Therefore, there is a need for a generic architecture that would support dynamic service deployment. The idea is having customized XML files that can be uploaded to an entity via a special service, and then this entity is able to auto-configure. Deployment information includes service definition, service implementation (can be anything beneath the service), device information, etc. An internal loader will parse the XML configuration file and generate or configure the entity and its services. Moreover, this approach can be extended for general configuration of agents, besides their services. An agent can be fully configured and re-configured with XML files that can be generated manually or due an export of specialized engineering tools (such as the Continuum Development Tools [12]). The approach is shown in Fig. 2.

A network connected device would also require announcing itself and discovering services in the system. Another technology part of the DPWS specification is the so called dynamic discovery. A recent connected device is able to announce itself by disperse messages into the network that can be captured by other entities. The same way, for locating a service, it will diffuse the search and wait for an answer. This contrast with centralized discovery mechanisms normally used in SOA, such as UDDI. In MAS there is a similar feature normally nicknamed as “yellow pages”. This is implemented by a specialized agent that can be used to store and retrieve information about agents and services. However, dynamic discovery does not need centralized discovery mechanism, thus is towards a more distributed agent system.

In terms of approaching to a MAS, agents could be easily configured, discovered and integrated into the system. Plug & play capabilities of agents can therefore introduce reduced development and maintenance efforts in the cases of re-configuration. Since plug & play is a well known requirement in modern

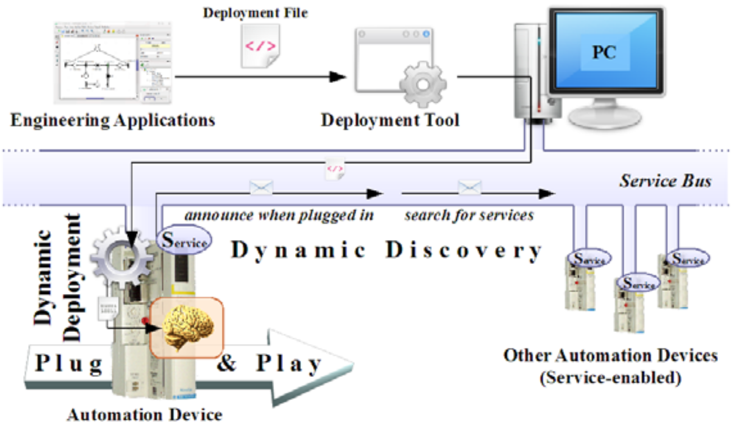


Fig. 2. Dynamic deployment, dynamic discovery and plug & play of service-oriented agents embedded into industrial controllers

distributed systems, removing an entity from the system is not less important. This can be understood as the mechanism of "plug, play & unplug".

2.4 Interactivity

Services are here to facilitate the integration of distributed systems and as a mean of interaction. Interaction between service providers and requesters can be partially archived by the aggregation of services based on predefined interactive rules. Such compositions are based on two well known styles, namely choreography and composition [13]. From the agents side, there are three main reasons why their actions need to be coordinated: dependencies between agents' actions, need to meet global constraints and insufficient competence of individual, resources or information to solve the entire problem [14]. The coordination can be achieved when agents are service-oriented and use the known techniques.

Recent experiments in service-oriented industrial automation were done with BPEL (Business Process Execution Language) and Petri net formalism (see [15] and [16]) to take coordinative actions for automation entities in the system. Moreover, in [13] a model-driven approach is given to design interoperable holonic multi-agent systems in a SOA, providing a definition of composition of services in these systems. Petri nets were used as a unified and integrated methodology. These nets are applied to the modeling, analysis, service management, embedded software controllers, decision support system and monitoring, to improve the fundamentals in the engineering of service-oriented automation systems. Agent-like service-oriented entities were the host of Petri net-based control with embedded decision support system to mediate automation devices [16].

2.5 Integration

To simplify the service integration into large-scale enterprise software, many large vendors have announced architectures and products for an enterprise service bus designed to facilitate the integration of enterprise software components [10]. With the advent of web service and SOA, realizing enterprise integration, accelerating enterprise responsiveness to customers, automating inter-enterprise interactions and optimizing the business processes of the whole supply chain become feasible. The service-oriented paradigm and the web service technologies are rapidly emerging as the most practical approaches for integrating a wide array of manufacturing resources in the manufacturing grid environment. Efforts in the semantic web standards and technologies present an opportunity for automating the integration process [17]. The mature use of SOA in business and e-commerce and the standardized information exchange is seen as a high-valuable component in the integration of service-oriented MAS into a cross layer network.

3 Conclusions and Future Work

Service-oriented injected multi-agent system is a current research topic. In industrial automation were MAS and SOA are being in continuous research, the fusion of both have been seen as a logical step. Therefore, this paper demonstrated these concepts, extracted from the research work in distributed industrial automation. From the previous sections, it can be concluded that MAS can only benefit with the enrichment of service features, not only for the communication, but also as a mean of orientation and development of these entities. Moreover, distribution and subsequent integration of parts are achieved more consistently with service-oriented multi-agent systems: from one side MAS is linked to autonomy, problem resolution intelligence and coordination of entities; SOA can boost those systems with uniform descriptions, communication platform, service-orientation as a philosophy for agents, seamless integration and well known coordinative processes. All these topics are important for future industrial automation and adjacent systems.

An outlook for research in this domain can be taken in different fronts. More industrial applications are needed to demonstrate and enforce future adoptions. For that, topics of performance, available software tools and efficiency have to be verified. Moreover, since it represents a new form of engineering contrasting the traditional IEC 61131, efforts have to be done in educational movements.

Acknowledgments. The authors would like to thank the European Commission and the partners of the EU IST FP6 project "Service-Oriented Cross-layer infrastructure for Distributed smart Embedded devices" (SOCRADES) and the European ICT FP7 project "Cooperating Objects Network of Excellence" (CONET) for their support.

References

1. Zeeb, E., Bobek, A., Bohn, H., Golatowski, F.: Lessons learned from implementing the Devices Profile for Web Services. In: IEEE International Conference on Digital Ecosystems and Technologies (2007)
2. Mehrabi, M.G., Ulsoy, A.G., Koren, Y.: Reconfigurable manufacturing systems: Key to future manufacturing. J. of Intelligent Manufacturing 11(4), 403–419 (2000)
3. Mick, R., Polsonetti, C.: Collaborative automation: The platform for operational excellence. White paper, ARC Advisory Group (2003)
4. Nguyen, X.T., Kowalczyk, R.: Enabling agent-based management of web services with WS2JADE. In: 5th International Conference on Quality Software, pp. 407–412 (2005)
5. Huhns, M.: Agents as web services. IEEE Internet Computing 6(4), 93–95 (2002)
6. Paulino, H., Lopes, L.: A service-oriented language for programming mobile agents. In: Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 1294–1296. ACM Press, New York (2006)
7. Mendes, J.M., Leitão, P., Restivo, F., Colombo, A.W.: Service-oriented agents for collaborative industrial automation and production systems. In: Mařík, V., Strasser, T., Zoitl, A. (eds.) HoloMAS 2009. LNCS, vol. 5696, pp. 13–24. Springer, Heidelberg (2009)
8. Shafiq, M.O., Ding, Y., Fensel, D.: Bridging multi agent systems and web services: towards interoperability between software agents and semantic web services. In: 10th IEEE International Enterprise Distributed Object Computing Conference, pp. 85–96 (2006)
9. OASIS. Devices Profile for Web Services, DPWS (2009), http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01
10. Bichier, M., Lin, K.J.: Service-oriented computing. Computer 39(3), 99–101 (2006)
11. Delamer, I., Martinez Lastra, J.: Ontology modeling of assembly processes and systems using semantic web services. In: IEEE International Conference on Industrial Informatics, August 2006, pp. 611–617 (2006)
12. Mendes, J., Bepperling, A., Pinto, J., Leitão, P., Restivo, F., Colombo, A.: Software methodologies for the engineering of service-oriented industrial automation: The Continuum project. In: Proc. 33rd Annual IEEE International Computer Software and Applications Conference, July 20-24, vol. 1, pp. 452–459 (2009)
13. Hahn, C., Fischer, K.: Service composition in holonic multiagent systems: Model-driven choreography and orchestration. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS (LNAI), vol. 4659, pp. 47–58. Springer, Heidelberg (2007)
14. Jennings, N.: Coordination techniques for distributed artificial intelligence (1996)
15. Puttonen, J., Lobov, A., Martinez Lastra, J.L.: An application of BPEL for service orchestration in an industrial environment. In: Proc. IEEE International Conference on Emerging Technologies and Factory Automation, pp. 530–537 (2008)
16. Mendes, J.M., Leitão, P., Colombo, A., Restivo, F.: High-level Petri nets control modules for service-oriented devices: A case study. In: Proc. 34th Annual Conference of IEEE Industrial Electronics, November 10-13, pp. 1487–1492 (2008)
17. Zhao, Y.Z., Zhang, J.B., Zhuang, L., Zhang, D.: Service-oriented architecture and technologies for automating integration of manufacturing systems and services. In: 10th IEEE Conference on Emerging Technologies and Factory Automation, vol. 1, pp. 349–355 (2005)

Design of a Neural Network for an Identification of a Robot Model with a Positive Definite Inertia Matrix

Jakub Możaryn and Jerzy E. Kurek

Institute of Automatic Control and Robotics, Warsaw University of Technology,
02-525 Warszawa, św.Andrzeja Boboli 8, Poland
J.Mozaryn@mchtr.pw.edu.pl, jkurek@mchtr.pw.edu.pl
http://jakubmozaryn.ovh.org

Abstract. This article presents a method of designing the neural network for the identification of the robot model in a form of *Lagrange-Euler* equations. It allows to identify the positive definite inertia matrix. A proposed design of a neural network structure is based on the *Cholesky* decomposition.

1 Introduction

A robot mathematical model is nonlinear and it has multiple-input-multiple-output (MIMO) structure. It can be calculated based on the *Lagrange-Euler* equations [2]. However, calculation of the model of the real industrial robot requires the knowledge of the exact values of the robot's kinematical and dynamical parameters that are hard to obtain [5].

There is a growing interest in the identification of the robot model using neural networks [7,8]. Its advantages are an approximation of the multivariable nonlinear functions, an easy adaptation of the model parameters and a very rapid calculation of the model equations, which are important for their potential suitability e.g. for a robot control system sythesis. Designing neural networks does not require an exact knowledge of the functions and the physical parameters that describe the model, but only the values of the model variables: generalized coordinates and control signals. Moreover the structure of the neural network can resemble the mathematical model of the robot and therefore the elements in the *Lagrange-Euler* equation can be distinguished.

However, there are some some drawbacks of this approach. Even though there are some potentially good results, usually the structure of the neural network does not satisfy the very strict properties that characterize the mathematical model of the robot. The most important one is a positive definiteness of an identified inertia matrix. Lack of this property can result e.g. in unstable control system.

In this paper we propose new design method of the neural network structure based on the *Cholesky* decomposition [6] where identified inertia matrix is positive definite.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 321–328, 2010.
© Springer-Verlag Berlin Heidelberg 2010

2 Discrete Time Robot Model

A discrete time model of the robot with n degrees of freedom, based on *Lagrange-Euler* equations can be presented as follows [2]

$$M(q,k)\Gamma(k) + P(q,k) = \tau(k) \ , \tag{1}$$

where

$$\Gamma(k) = [\gamma_i(k)] = T_p^{-2}[q(k+1) - 2q(k) + q(k-1)] \ , \tag{2}$$

$$P(k) = V(q,k) + G(q,k) \tag{3}$$

and $q(k) = [q_i(k)] \in R^n$ is a vector of the generalized joint coordinates, $M(q,k) = [m_{i,j}(k)] \in R^{n \times n}$ a robot inertia matrix, $V(q,k) = [v_i(k)] \in R^n$ a vector of *Coriolis* and centrifugal forces, $G(q,k) = [g_i(k)] \in R^n$ a vector of a gravity forces, $\tau(k) = [\tau_i(k)] \in R^n$ a vector of control signals, k a discrete time, T_p a sampling period ($t = kT_p$).

3 Properties of the Robot Model

There are specific properties [2,3,7] of the robot mathematical model, especially regarding the inertia matrix $M(k)$.

Property 1. The inertia matrix $M(k)$ is symmetrical

$$M(q,k) = M^T(q,k) \ . \tag{4}$$

Property 2. The inertia matrix $M(k)$ is positive definite.

$$\forall_{x \in R^n, x \neq 0}, \ x^T M(q,k) x > 0 \ , \tag{5}$$

$$\det[M(q,k)] \neq 0 \ . \tag{6}$$

Property 3. Each element of the inertia matrix $M(k)$ is a nonlinear function described as

$$m_{i,j}(k) = f(q_{s+1}(k), ..., q_n(k)), \ s = \min(i,j) \ . \tag{7}$$

Properties 1 and 2 can be used to design the identification method of the inertia matrix elements. Third property defines the set of the generalized coordinates that influence each element of the inertia matrix. Especially the element $m_{n,n}(k)$ doesn't depend on the generalized coordinates and is constant.

4 Properties of the Positive Definite Inertia Matrix

One can transform inertia matrix as follows

$$\bar{M}(q,k) = \Lambda M(q,k) \Lambda^T = \begin{bmatrix} m_{n,n} & \cdots & m_{n,1} \\ \vdots & \ddots & \vdots \\ m_{1,n} & \cdots & m_{1,1} \end{bmatrix} \tag{8}$$

where

$$\Lambda = \begin{bmatrix} 0 \cdots 1 \\ \vdots \ \ . \ \ \vdots \\ 1 \cdots 0 \end{bmatrix} . \tag{9}$$

Note, that according to property 3 (7) element $\bar{m}_{1,1} = m_{n,n}$ is constant.

Then, using the *Cholesky* decomposition [6] construct an upper triangular matrix $U \in R^{n \times n}$ for the given positive definite symmetrical matrix $\bar{M}(q,k)$, such that

$$\bar{M}(q,k) = U^T U \tag{10}$$

and

$$U = \begin{bmatrix} u_{1,1} & \cdots & u_{n,1} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & u_{n,n} \end{bmatrix} . \tag{11}$$

Elements of the upper triangular matrix U are calculated as follows

$$u_{1,1} = \sqrt{\bar{m}_{1,1}} = \sqrt{m_{n,n}} \ , \tag{12}$$

$$u_{1,j} = \frac{1}{u_{1,1}} \bar{m}_{1,j} = \frac{1}{u_{1,1}} m_{n,n-j+1}, \ j = 2, \ldots, n \ , \tag{13}$$

$$u_{k,k} = \sqrt{\bar{m}_{k,k} - \sum_{i=1}^{k-1} u_{i,k}^2} = \sqrt{m_{n-k+1,n-k+1} - \sum_{i=1}^{k-1} u_{i,k}^2}, \ k = 2, \ldots, n \ , \tag{14}$$

$$\begin{gathered} u_{k,j} = \tfrac{1}{u_{k,k}} (\bar{m}_{k,j} - \textstyle\sum_{i=k+1}^{n} u_{i,k} u_{i,j}) = \tfrac{1}{u_{k,k}} (m_{n-k+1,n-j+1} - \textstyle\sum_{i=k+1}^{n} u_{i,k} u_{i,j}), \\ k = 1, \ldots, n, \ j = k+1, \ldots, n \ . \end{gathered} \tag{15}$$

It is easy to notice, that the matrix $\bar{M}(q,k)$ is positive definite iff $u_{i,i} > 0$ ($\det(U) \neq 0$). Thus, from (12)-(15), the following conditions are satisfied by the elements on a diagonal of the positive definite inertia matrix

$$m_{n,n} > 0 \ , \tag{16}$$

$$m_{n-k+1,n-k+1} > \sum_{i=1}^{k-1} u_{i,k}^2, \ k = 2, \ldots, n \tag{17}$$

5 Neural Network Robot Model

In the robot model (1) the unknown nonlinear elements of $M(q,k), P(q,k)$ should be identified. For their identification a feed forward neural network can be used.

Let us describe the neural network model for the identification of the model (1). We have assumed that the inputs to the neural network are $q(k-1), q(k)$, and $\Gamma(k)$ (2). The output of the neural network is vector $\tau(k)$. The neural network preserves the multiple-input-multiple-output structure of the robot model. Each

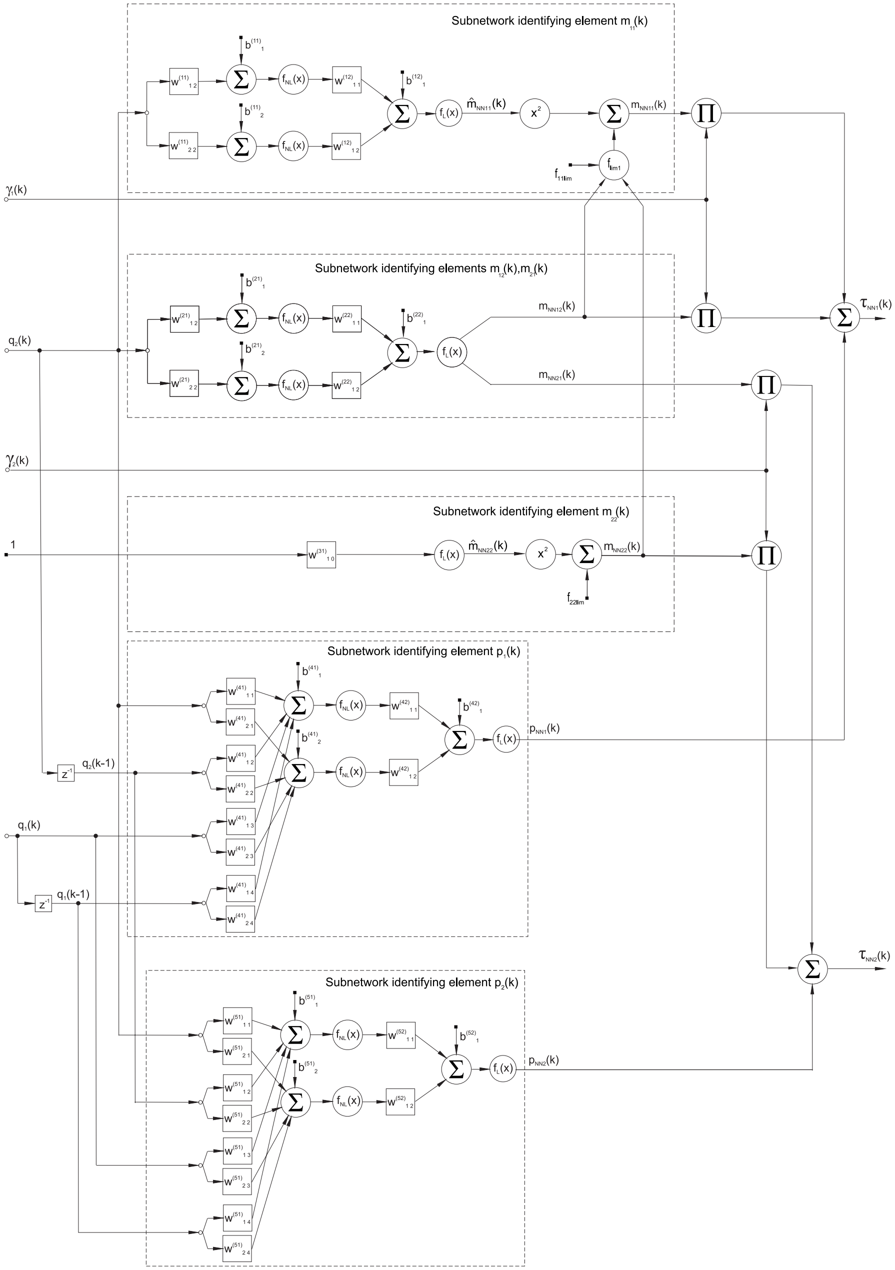


Fig. 1. A neural network for the identification of the mathematical model of robot with two degrees of freedom

element of $M(q,k)$, and $P(q,k)$ is identified by the distinct subnetwork. Inputs to each subnetwork are determined by property 3 (7). Elements $m_{i,j}(q,k)$ and $m_{j,i}(q,k)$, $i \neq j$ are identified by one subnetwork, therefore the identified matrix $M_{\mathrm{NN}}(k) = [m_{\mathrm{NN}i,j}] \in R^{n\times n}$ (estimate of inertia matrix) is symmetrical.

In all nonlinear layers neurons are described by a nonlinear activation function, e.g. sigmoidal

$$y = f_{nl}(x) = \mathrm{tansig}(x) \ . \tag{18}$$

In linear layers there are neurons described by a linear activation function

$$y = f_l(x) = x \tag{19}$$

where $x = \sum_{i=1}^{L} w_i x_i + b$, L is a number of neuron inputs, w_i a weight of the i-th input to neuron, x_i i-th input to neuron, b a threshold offset.

Element $m_{\mathrm{NN}n,n}$ is identified using one layer subnetwork with one neuron described by linear activation function (19) with the input that equals 1.

Elements on the diagonal of the matrix $M_{\mathrm{NN}}(k)$ are calculated as follows

$$m_{\mathrm{NN}i,i}(k) = \hat{m}^2_{\mathrm{NN}i,i}(k) + f_{\lim i}, \ i = 1,\ldots,n \tag{20}$$

where $\hat{m}_{\mathrm{NN}i,i}(k)$ is calculated by the subnetwork and $f_{\lim i}$ is a limit function bounding a value of the element $m_{\mathrm{NN}i,i}(k)$ in order to satisfy (16) and (17).

Let us present an example of the neural network for the identification of the robot with two degrees of freedom and the symmetrical positive definite inertia matrix. The limit functions of the elements on diagonal of identified matrix $M_{\mathrm{NN}}(k)$ are

$$f_{\lim 2} = f_{22\lim} \ , \tag{21}$$

$$f_{\lim 1} = u^2_{1,2} + f_{11\lim} = \frac{m_{NN1,2}(k)^2}{m_{NN2,2}(k)} + f_{11\lim} \tag{22}$$

where $f_{ii\lim} = \mathrm{const}, \ f_{ii\lim} > 0, \ i = 1,2$ are chosen arbitrarily.

In Fig. 1 it is presented an example of the neural network for the identification of the robot with two degrees of freedom and the symmetrical positive definite inertia matrix. Because the bounds (16) and (17) are satisfied the identified inertia matrix is positive definite. Moreover, the linear transformation (8) allows to obtain a constant value of $m_{\mathrm{NN}n,n}(k)$ and *Property 3* is satisfied.

6 Computer Simulations

In order to verify the proposed neural network robot model the data from the simulation of the robot with two degrees of freedom [3,8] was used. The physical parameters of robot are gathered in Table 1.

For calculation of the training and testing data a reference trajectory $q_{\mathrm{r}}(k) = [q_{\mathrm{r}i}(k)] \in R^2, \ i = 1,2$ for every joint was set as the sum of three different *sine* functions, according to the following formula

$$q_{\mathrm{r}i}(k) = \sum_{j=1}^{3} a_{ij} cos(\varpi_{ij} k T_p + \varphi_{ij}), \ i = 1,2 \tag{23}$$

Table 1. Physical parameters of a robot with two degrees of freedom assigned according to the *Denavitt-Hartenberg* notation [2]

Link i	$\alpha_i[^\circ]$	$a_i[m]$	$\theta_i[^\circ]$	$d_i[m]$
1	0	2.0	q_1	0
2	0	0.8	q_2	0
	M[kg]	$r_x[m]$	$r_y[m]$	$r_z[m]$
1	22	-1.0	0	0
2	16	-0.4	0	0
	$I_{xx}[kg \cdot m^2]$	$I_{yy}[kg \cdot m^2]$	$I_{zz}[kg \cdot m^2]$	$J_{xy}, J_{yz}, J_{xz}[kg \cdot m^2]$
1	0	22.00	22.00	0
2	0	2.56	2.56	0

where $i = 1, 2$ is a joint number, a_{ij} is an amplitude, ϖ_{ij} is an angular velocity, φ_{ij} is the phase. The values of a_{ij}, ϖ_{ij}, φ_{ij}, different for the training and testing trajectories, are given in Table 2.

Table 2. The parameters of the training and testing trajectories

Link i	$a_{i1}[^\circ]$	$a_{i2}[^\circ]$	$a_{i3}[^\circ]$	$\varpi_{i1}[\frac{\circ}{s}]$	$\varpi_{i2}[\frac{\circ}{s}]$	$\varpi_{i3}[\frac{\circ}{s}]$	$\varphi_{i1}[^\circ]$	$\varphi_{i2}[^\circ]$	$\varphi_{i3}[^\circ]$
Link 1, training	110	23	47	37	5	8	127	77	54
Link 2, training	30	26	124	30	5	15	97	27	125
Link 1, testing	61	92	27	38	6	6	89	161	147
Link 2, testing	104	55	21	29	6	15	52	61	96

For simulation we have chosen a time interval $T = 100[sec]$, with a sampling time $T_p = 0.01[sec]$. Therefore, there were $N = 10000$ data samples for training and $N = 10000$ data samples for testing of the neural network model. Every element of the robot mathematical model was identified by the subnetwork containing one nonlinear hidden layer with 2 neurons, and one linear output layer with 1 neuron. Neural networks were trained using the backpropagation method and the *Levenberg-Marquardt* method to update weights in all layers [4]. There were 50 training iterations. Bounding constants in (21) and (22) were chosen arbitrarily as

$$f_{ii\,\lim} = 0.001, \; i = 1, 2 \; . \tag{24}$$

The performance function of the neural network was chosen as a mean squared error

$$J = \frac{1}{N} \sum_{k=1}^{N} \sum_{i=1}^{2} [\tau_i(k) - \tau_{\mathrm{NN}i}(k)]^2 \; . \tag{25}$$

The neural network model of the robot was trained off-line with the known training trajectory to identify the model coefficients. Afterwards it was tested for estimation of the testing data. The positive definiteness of the real matrix was verified using the *Sylvester* criterion [1,6]. During simulations the identified inertia matrix was always positive definite.

One of the applications of presented neural networks is the position estimation, which can be done according to the following formula

$$q_{\mathrm{NN}}(k+1) = T_p^2 M_{\mathrm{NN}}(k)^{-1}[\tau(k) - P_{\mathrm{NN}}(k)] + 2q(k) - q(k-1)\ . \qquad (26)$$

Matrix $M_{\mathrm{NN}}(k)$ is positive definite and matrix $M_{\mathrm{NN}}(k)^{-1}$ always exists. Therefore $q_{\mathrm{NN}}(k+1)$ can be easily calculated. The accuracy of the position estimation was checked using the *average absolute position errors* of an end effector in a cartesian coordinate system of a robot base.

$$e_{\mathrm{avOX}} = \frac{\sum_{k=1}^{N} |x_{\mathrm{r}}(k) - x_{\mathrm{NN}}(k)|}{N}, \qquad (27)$$

$$e_{\mathrm{avOY}} = \frac{\sum_{k=1}^{N} |y_{\mathrm{r}}(k) - y_{\mathrm{NN}}(k)|}{N}. \qquad (28)$$

where $\{x_{\mathrm{r}}(k), y_{\mathrm{r}}(k)\}$ is the reference position, $\{x_{\mathrm{NN}}(k), y_{\mathrm{NN}}(k)\}$ is the estimated position of the end effector.

Obtained values of the errors (27), (28) for training and testing data are presented in Table 3. There are compared errors calculated for the neural networks before and after training.

Table 3. The values of the average absolute position errors e_{avOX}, e_{avOY} for training and testing trajectories

before training				after training			
training trajectory		testing trajectory		training trajectory		testing trajectory	
e_{avOX}[m]	e_{avOY}[m]	e_{avOX}[m]	e_{avOY}[m]	e_{avOX}[m]	e_{avOY}[m]	e_{avOX}[m]	e_{avOY}[m]
0.40	0.36	0.53	0.55	0.02	0.02	0.03	0.03

Obtained results, presented in Table 3, show that position estimation improves after training around 20 times for training and testing data. However one can expect the position estimation errors below 0.1 [mm], therefore obtained values (around 20-30 [mm]) presented in the example can be still too big for some applications.

7 Concluding Remarks

This paper presents the design method of the neural network for the identification of the robot mathematical model with the positive definite inertia matrix. The proposed, new method is based on the conditions that are obtained from the *Cholesky* decomposition of an estimated inertia matrix. Simulations of identification of the model of the robot with two degrees of freedom show that the proposed method assures meeting the positive definiteness condition. However, there are certain problems that should be solved in the future.

In general the identified inertia matrix is positive definite, but the values on its diagonal are bounded by constant values that are chosen arbitrarily. Therefore further research should be done to find a methodology of the limiting values choice e.g. in an adaptive manner. An important issue is also usefulness of the presented method in a robot control system [8]. Moreover, the estimated inertia matrix can be used in other applications where the inversion of inertia matrix is required e.g. estimation of robot position. However, presented results indicate that there are still errors inappropriate for some applications. Therefore, some methods of improvement of neural network structure and numerical algorithms should be considered [9].

References

1. Johnson, C.R.: Positive Definite Matrices. Amer. Math. Monthly 77, 259–264 (1970)
2. Fu, K.S., Gonzalez, R.C., Lee, C.S.G.: Robotics: control, sensing, vision and inteligence. McGraw-Hill Book Company, New York (1987)
3. Tang, K.M.W., Tourassis, V.D.: Systematic simplification of dynamic robot models. In: Proc. Midwest Symp. Circuits Syst., Syracuse, NY, pp. 1031–1034 (1987)
4. Osowski, S.: Neural networks for information processing. OWPW, Warsaw (1994) (in Polish)
5. Corke, P.I., Armstrong-Hélouvry, B.: A search for consensus among model parameters reported for the PUMA 560 robot. In: Proc.IEEE Int. Conf. Robotics and Automation, San Diego, vol. 1, pp. 1608–1613 (1994)
6. Golub, G.H., Van Loan, C.F.: Matrix computations, 3rd edn. Johns Hopkins University Press, Baltimore (1996)
7. Lewis, F.L.: Neural network control of robot manipulators and nonlinear systems. Taylor & Francis, Abington (1999)
8. Możaryn, J., Kurek, J.E.: Synthesis of sliding mode control of robot with neural network model. In: Proc. 12th IEEE Int. Conf. on Methods and Models in Automation and Robotics, MMAR 2006, vol. 2 (2006)
9. Możaryn, J., Kurek, J.E.: Using Tikhonov regularization to improve estimation of robot position based on uncertain robot model obtained by neural network. Pomiary Automatyka Kontrola 3 (2009)

A Fast Image Analysis Technique for the Line Tracking Robots

Krzysztof Okarma and Piotr Lech

West Pomeranian University of Technology, Szczecin
Faculty of Electrical Engineering,
Chair of Signal Processing and Multimedia Engineering,
26. Kwietnia 10, 71-126 Szczecin, Poland
{krzysztof.okarma,piotr.lech}@zut.edu.pl

Abstract. Fast and simplified image processing and analysis methods can be successfully implemented for the robot control algorithms. Statistical methods seem to be very useful for such an approach, mainly because a significant reduction of analysed data is possible. In the paper the use of the fast image analysis based on the Monte Carlo area estimation for the simplified binary representation of the image is analysed and proposed for the mobile robot control. A possible implementation of the proposed method can applied in the line tracking robots and such application has been treated as the basic one for the testing purposes.

Keywords: robot vision, statistical image analysis, line tracking robots.

1 Technical Background

Line tracking is one of the most relevant aspects of control systems designed for mobile robots. A typical implementation of the optical part of the line following robot is based on a line of sensors gathering information about the position of the traced line, usually underneath the robot. A line sensor contains a number of cells determining the resolution of the optical system. Another important parameter is the distance from the sensors to the centre of steering, responsible for the maximum speed of the properly controlled robot. For the specified number of cells in the line sensor, the spacing between them is also important for the smoothness of the robot's motion.

One of the main disadvantages of such optical systems is the short time for the reaction limited by the distance between the sensors and the steering centre (or wheels). Another one is relatively low resolution of the tracking system. A significant increase of the resolution, together with wide possibilities of analysing data acquired from the front, have caused the interest of using the vision based systems for that purposes [2,6]. Nevertheless the analysis of the full image even with low resolution is time consuming and computationally demanding, especially assuming varying lighting conditions, the presence of some obstacles, some line intersections etc.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 329–336, 2010.
© Springer-Verlag Berlin Heidelberg 2010

The proposed solution, based on the fast image analysis using the Monte Carlo approach, allows to reduce the amount of processed data preserving the main advantages of the vision systems.

2 The Fast Monte Carlo Image Analysis

The image analysis algorithm used in the paper is based on the idea of a fast image processing methods dedicated to the real-time systems supplied by the Monte Carlo method [7]. Differently than in the primary version of the algorithm, the camera moves relatively to the static scene but the working properties of the method are similar.

2.1 The Image Binarization Algorithm

The processing algorithm for the images acquired from the video camera consists of the following steps:

- each frame is divided into $T \times S$ square blocks of $r \times r$ pixels,
- for each block the area of the object is estimated using the Monte Carlo method,
- the binary array is constructed using the estimated area values obtained in the previous step, where the following values are set:
  - 0 – for the estimated object's area within the corresponding block below the specified threshold,
  - 1 – for the estimated object's area within the corresponding block above the specified threshold.

In order to filter the undesired data, usually related to some contaminations and not used for the robot control purposes, the binarization threshold is set. In the experiments illustrated in the Fig. 1 the additional "cut-off" value has been set as equivalent to minimum 20% black pixels within the block in order to avoid the influence of the small artefacts, especially close to the region of interest. As the final result the simplified binary representation of the image is obtained where such artefacts (the elements which do not represent the line) have been removed during the "cut-off" operation.

The proper choice of the parameters T and S, determining the number of the analysed binary blocks, is the compromise between the processing speed and its precision.

2.2 The Area Estimation Using the Monte Carlo Method

Performing a statistical experiment with the use of the Monte Carlo method [7], an algorithm for the estimation of the pixels fulfilling a specified logical criterion in the image can be proposed. The major advantage of counting the pixels drawn using a pseudo-random generator is the reduction of the number of pixels used for the analysis, so the overall performance of the algorithm can be increased significantly.

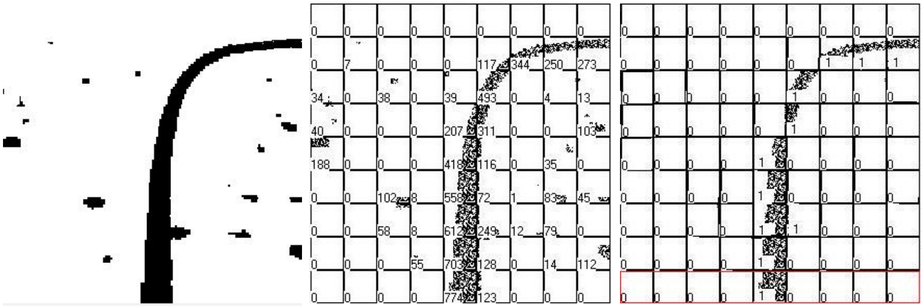


Fig. 1. The visualisation of a curve after the preliminary Monte Carlo image analysis (the original binary image, the frame divided into blocks with estimated area and the filtered image after "cut-off" operation)

The samples from the binary image are numbered and stored in the one-dimensional vector, where "1" stands for the black pixels and "0" denotes the white ones. In the next step a single element from the vector is randomly chosen, which is then returned, so n independent draws are performed. To simplify the theoretical analysis only a single moving dark object (Ob) in the scene (Sc) with a light constant background is assumed (a similar situation takes place in some typical line following scenarios).

Dividing the scene into squares, the K_N squares from N elements of the scene represent the object, so the probability of choosing the point representing the object is equal to:

$$p = \frac{K_N}{N} \tag{1}$$

The geometrical probability of choosing the point on the object's surface for the infinite number of samples is given as:

$$p = \lim_{N \to \infty} \frac{K_N}{N} = \frac{A_{Ob}}{A_{Sc}} \tag{2}$$

so the area of the object can be estimated as:

$$A_{Ob} \approx K_N \cdot \frac{A_{Sc}}{N} \tag{3}$$

The area of the scene is equal to $A_{Sc} = N \cdot k_x \cdot k_y$ (where k_x and k_y are the scale factors for x and y coordinates), so:

$$A_{Ob} = K_N \cdot \frac{N}{N} \cdot k_x \cdot k_y = K_N \cdot k_x \cdot k_y \tag{4}$$

The above analysis allows using the probability of choosing the point representing the object in the proposed algorithm with a reduced number of analysed samples. In this purpose a statistical experiment using the Monte Carlo method can be used. The basis of this method are two statements: the law of large numbers and the central limit theorem. In the first one it is stated that for the sequence of successive approximations of the estimated value is convergent to the sought solution and the second one shows how far from the solution is the actual value after performing the given number of statistical tests.

The luminance samples after the binarization (being "ones" or "zeros" corresponding to the allowed values of the specified logical condition) stored in one-dimensional vector are chosen randomly, so for the single draw a random variable X_i of the two-way distribution is obtained, what leads to the following probability expressions:

$$P(X_i = 1) = p \qquad P(X_i = 0) = q \tag{5}$$

where $p + q = 1$, $E(X_i) = p$, $V(X_i) = p \cdot q$.

The logical condition should be specified in the way allowing to distinguish the samples representing the object from the background. In our application a constant background can be assumed as well as the vehicles' colours differing noticeably from the background (the motion detector can be also helpful).

For n draws the variable Y_n is obtained:

$$Y_n = \frac{1}{n} \cdot \sum_{i=1}^{n} X_i \tag{6}$$

Using the Lindberg-Levy's theorem, the distribution of Y_n tends to the normal distribution $N(m_y, \sigma_y)$ if $n \to \infty$. Calculating the expected value of Y_n as $E(Y_n) = p$ and variance $V(Y_n) = \frac{p \cdot q}{n}$ it can be stated that the random value Y_n has the normal distribution with the following parameters:

$$m_y = p \tag{7}$$

$$\sigma_y = \sqrt{\frac{p \cdot q}{n}} \tag{8}$$

Because of the asymptotic normal distribution $N(p, \sqrt{p \cdot q/n})$ it can be noticed that for the variable Y_n the central limit theorem is fulfilled.

Using the substitution:

$$U_n = \frac{Y_n - m_y}{\sigma_y} \tag{9}$$

the normal distribution can be standardized and marked as the standard normal distribution $N(0, 1)$.

In the interval estimation method the following formula is used:

$$p(|U_n| \leq \alpha) = 1 - \alpha \tag{10}$$

Assuming the interval:

$$|U_n| \leq u_\alpha \tag{11}$$

considering also the expressions (7), (8) and (9), the following expression is obtained:

$$\left| Y_n - \frac{K_N}{N} \right| \leq \varepsilon_\alpha \tag{12}$$

where

$$\varepsilon_\alpha = \frac{u_\alpha}{\sqrt{n}} \cdot \sqrt{\frac{K_N}{N} \cdot \left(1 - \frac{K_N}{N}\right)} \tag{13}$$

The estimator of the probability p (eq. 1) for k elements from n draws, can be expressed as:

$$\hat{p} = \frac{k}{n} = \frac{\sum_{i=1}^{n} X_i}{n} = Y_n \tag{14}$$

and the estimator of the object's area is given as the following formula:

$$\hat{A}_{Ob} = \hat{p} \cdot A_{Sc} = Y_n \cdot A_{Sc} = \frac{k}{n} \cdot A_{Sc} \tag{15}$$

Using the equations (12) and (15) the formula describing the interval estimation for the object's area is obtained as:

$$\left| \frac{\hat{A}_{Ob}}{A_{Sc}} - \frac{K_N}{N} \right| \leq \varepsilon_\alpha \tag{16}$$

where ε_α is given by the equation (13).

The considerations presented above are correct for a random number generator with the uniform distribution. Such a generator should have good statistical properties. Presented algorithms are identical to the methods of the 2-D object's area estimation expressed in pixels. Nevertheless, their applicability is limited by their integrating character.

3 A Simple Proportional Control Algorithm for a Line Following Robot Control

The mobile robot control process presented in the paper is based on the popular proportional control approach used for controlling the robots with sensors based on the infra-red receivers grouped into the line containing a specified number of the infra-red cells. In our approach the differential steering signals for the motors are obtained using the first (the lowest) row of the simplified binary image which is equivalent to the typical line of the infra-red sensors. The idea of the differential motion controlling is illustrated in Fig. 2.

The algorithm presented in the paper can be disturbed by some artefacts caused by the errors during the preprocessing step (e.g. insufficient filtration of some undesired elements on the image). The optimal situation takes place for a single "one" in the lowest row of the image. In the simplified version of the algorithm it can be assumed that there are no line crossings. The reduction of the possible disturbances can be achieved using the following additional operations:

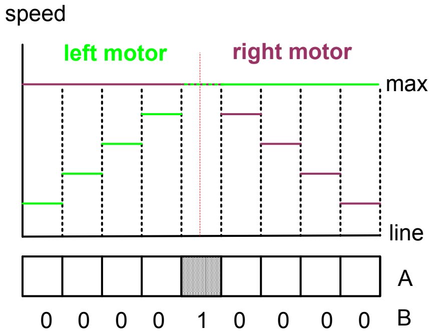


Fig. 2. A proposal of simple control: A – the lowest row of the binary image, B - the binary representation of the blocks, C – control signal for the element “1”

- limitation of the line thickness by the proper choice of the binarization threshold (lines containing more than a single ”one” can be ignored assuming no line crossings),
- orphan blocks removal (applicable for all blocks with the value ”one” without any neighbouring ”ones”),
- ignoring the empty lines (without any ”ones”).

4 Extension of the Robot Control by Using the Image Based Prediction

The control algorithm of the robot tries to lead to the situation when the line fills only a single block of the bottom image row. In all the cases when the line occupies the two neighbouring blocks the choice of the block is based on the simple analysis of the current trend. It is assumed that the turn angles of the line match the motion and control possibilities of the robot. The main element of the modified control system is the variable (flag), computed using the previous frames, which informs the system about the current state (0 – moving forward or turning, 1 – turning with detected crossing lines).

The control algorithm for a single frame can be described as follows:

- binarization of the current frame,
- orphan blocks removal,
- filling the gaps in the detected line using the approximation methods,
- control operation:
  - the detection of the crossing lines
    - ∗ if not turning: moving forward with the maximum speed for the symmetrical line-crossing (Fig. 3e) or using the speed control
    - ∗ else: turning and using the speed control if the maximum value is not in the middle block (Fig. 3f)
  - speed control: speed is proportional to the sum of the values in the middle blocks of each horizontal line (if zero, the minimum speed is set before the turning and the control flag is set to 0) - see Fig. 3g and 3h.

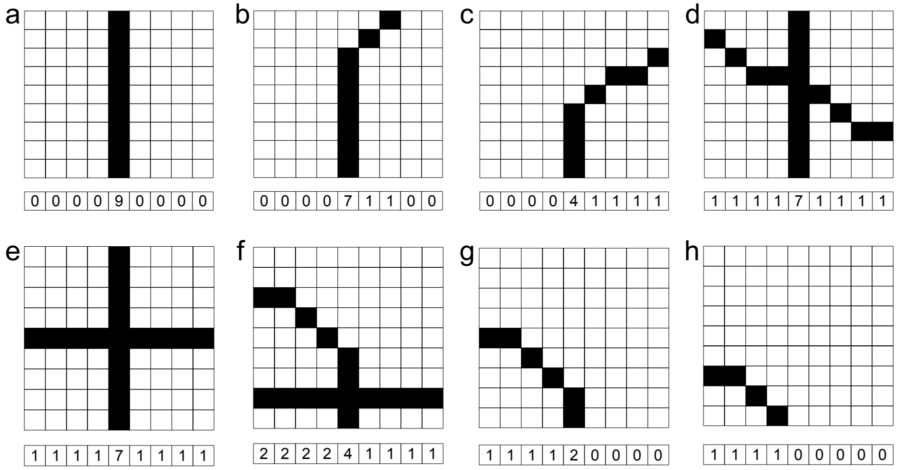


Fig. 3. The illustration of the image based prediction and control (a - forward with maximum speed, b - fast forward and turn, c - forward and turn, d - crossing, e - crossing, f - crossing and turn, g - slow forward and turn h - start turning)

5 Conclusions

Observing the working properties of the proposed algorithm, the oscillating character of the path for the robot reaching the specified line can be noticed, what is a typical feature of the proportionally controlled systems, causing the characteristic oscillations of the images acquired from the camera. Performed experiments have verified the usefulness of the fast image analysis based on the Monte Carlo method for the controlling of the line following robot.

Comparing to some other more advanced statistical methods [3] e.g. popular Markov Chain Monte Carlo [9] or Sequential Monte Carlo [1,8], which can be used for the image analysis purposes, the approach proposed in the paper is a simplified but very efficient method, especially useful for the real-time applications. The main advantages of the proposed method is its low computational complexity causing the relatively high processing speed, especially in the real-time embedded systems. An application for the control of the mobile robot presented in the paper, despite of the simplicity of its control algorithm, is comparable with some typical ones based on the classical approach (proportional steering based on the infra-red sensors). Nevertheless, its advantage is the possibility of using some prediction algorithms allowing the increase of the robot's dynamic properties and fluent changes of its speed depending on the shape of the line in front of the robot.

The future work can be related to the extension of the control algorithm into the PID and some experiments with the prediction algorithms as well as the application of the control algorithm in the dynamic real-time system. Similar approaches can be also successfully applied for many other purposed based on the image and video processing e.g. fast image and video quality estimation [5] or extraction of geometrical features [4].

References

1. Chen, D., Odobez, J.-M.: Sequential Monte Carlo Video Text Segmentation. In: International Conference on Image Processing, ICIP 2003, vol. 3, pp. 21–24. IEEE Press, New York (2003)
2. Dupuis, J., Parizeau, M.: Evolving a Vision-Based Line-Following Robot Controller. In: 3rd Canadian Conference on Computer and Robot Vision, June 7-9, pp. 75–75 (2006), doi:10.1109/CRV.2006.3
3. Fearnhead, P.: Computational Methods for Complex Stochastic Systems: A Review of Some Alternatives to MCMC. Statistics and Computing 18(2), 151–171 (2008)
4. Okarma, K., Lech, P.: Monte Carlo Based Algorithm for Fast Preliminary Video Analysis. In: Bubak, M., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2008, Part I. LNCS, vol. 5101, pp. 790–799. Springer, Heidelberg (2008)
5. Okarma, K., Lech, P.: A Statistical Reduced-Reference Approach to Digital Image Quality Assessment. In: Bolc, L., Kulikowski, J.L., Wojciechowski, K. (eds.) ICCVG 2008. LNCS, vol. 5337, pp. 43–54. Springer, Heidelberg (2009)
6. Rahman, M., Rahman, M.H.R., Haque, A.L., Islam, M.T.: Architecture of the Vision System of a Line Following Mobile Robot Operating in Static Environment. In: 9th International Multitopic Conference, IEEE INMIC 2005, December 24-25, pp. 1–8 (2005), doi:10.1109/INMIC.2005.334473
7. Rubinstein, R.Y.: Simulation and the Monte Carlo Method. Wiley, Chichester (1981)
8. Vermaak, J., Ikoma, N., Godsill, S.J.: Sequential Monte Carlo Framework for Extended Object Tracking. IEE Proc. Radar Sonar Navig. 152(5), 353–363 (2005)
9. Zhai, Y., Shah, M.: Video Scene Segmentation Using Markov Chain Monte Carlo. IEEE Trans. on Multimedia 8(4), 686–697 (2006)

Multi-agent Logic with Distances Based on Linear Temporal Frames[*]

Vladimir Rybakov and Sergey Babenyshev

Department of Computing and Mathematics,
Manchester Metropolitan University,
John Dalton Building, Chester Street, Manchester M1 5GD, U.K.
V.Rybakov@mmu.ac.uk, S.Babenyshev@mmu.ac.uk

Abstract. The paper investigates a new temporal logic $\mathcal{TL}^M_{Dist}$, which combines temporal operations with the operations of localised agent's knowledge and operations responsible for measuring distances. The main goal is to construct a logical framework for modelling logical laws, which describe interactions between such operations. We consider issues of satisfiability and decidability for $\mathcal{TL}^M_{Dist}$. Our principal result is the algorithm which recognizes theorems of $\mathcal{TL}^M_{Dist}$, which implies that $\mathcal{TL}^M_{Dist}$ is decidable, and the satisfiability problem for $\mathcal{TL}^M_{Dist}$ is solvable.

Keywords: multi-agent systems, multi-modal logics, decision algorithms, satisfiability, Kripke semantics, distance measuring.

1 Introduction

This paper attempts to simulate evolutions of a distributed system of agents in a logical framework, based on linear temporal frames with distances. Temporal logics were first suggested to specify properties of programs in the late 1970's (cf. Pnueli [1]). The most used temporal framework is linear discreet-time frames, which have been extensively studied from the point of view of various prospects of applications (cf. e.g. Manna and Pnueli [2], Barringer, Fisher, Gabbay and Gough [3]). Model checking for such frames formed a strong direction in Logic in Computer Science, which uses, in particular, automata theory (cf. Vardi [4]). Temporal logics themselves can be considered as a special cases of hybrid logics, e.g. as bimodal logics with some laws imposed on the interaction of modalities to imitate the flow of time.

The mathematical theory devoted to the study of axiomatizations of temporal logics and development of their semantic theory based on Kripke/Hintikka-like models and temporal Boolean algebras formed a highly technical branch in the area of Non-classical logic (cf. van Benthem [5,6], Gabbay and Hodkinson [7], Hodkinson [8]). Axiomatizations of various (uni)-temporal linear logics are summarized in de Jongh et al. [9].

[*] This research is supported by Engineering and Physical Sciences Research Council (EPSRC), UK, grant EP/F014406/1.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 337–344, 2010.
© Springer-Verlag Berlin Heidelberg 2010

The introduction of agents into the picture, brings additional challenges for the researcher looking to simulate such systems. One of the problems is that introducing agents might lead to undecidability (cf. Kacprzak [10]). Another problem is that, non withstanding some previous successful attempts to apply multi-modal propositional logics toward the task of describing evolutions of multi-agent systems (cf. Fagin et al. [11]), there are still multiple technical problems remain, some of mathematical and some of conceptual nature. For instance, picking out only discreet, already accomplished states of affairs, prevents from looking into the structure of the transition process, which would be the main interest from the simulation point of view.

To bridge this gap, at this paper we introduce a new temporal logic $\mathcal{TL}^{M}_{Dist}$, which describes frames where transition periods are filled with intermediate states. This construction is necessarily based on some tools of general fusion theory. The issues of satisfiability and decidability for $\mathcal{TL}^{M}_{Dist}$ is our main interest: we find an algorithm which recognizes theorems of logic $\mathcal{TL}^{M}_{Dist}$ (which implies that it is decidable, and the satisfiability problem for it is decidable). The algorithm works as follows: an arbitrary formula in the language of $\mathcal{TL}^{M}_{Dist}$ is, first, transformed into a rule in a special normal reduced form, which, then, is checked for validity on special models (of size efficiently bounded in the size of the rule) with respect to special kind of valuations.

The general methodology of this paper is borrowed from [12] and [13].

2 Language and Semantics of $\mathcal{TL}^{M}_{Dist}$

The main goal of this paper is to understand how one can "fill in" the intervals in seemingly discreet temporal moments. In other words, to answer the question: "Can we simulate by finitary means how we arrive at the (next) discreet moment-state of affairs?"

The way we chose to investigate, is to suppose that there is a finite chain (may be more than one) of intermediate events, reached by interaction of agents, that starts late "Today" and ends up as "Tomorrow". This somewhat reminds the software development cycles, where stable releases are "interspersed" with alpha or beta builds (each of them can be "promoted" to the stable status or on the contrary outright abandoned).

We start from the formal definition of the logical syntax.

Language of $\mathcal{TL}^{M}_{Dist}$ To build logical language, we add to usual temporal operations Next (next) and Until (until), new unary logical operations K_i for agents' knowledge, a special operator Today, together with a countable set of operations for measuring temporal distances $\{\Diamond^{+}_{k}\}_{k\in\mathbb{N}}$.

Altogether, propositional language $\mathcal{L}$ includes (logical connectives are given with their arities as upper-right indices):

$$\mathcal{L} := \langle \vee^2, \wedge^2, \to^2, \neg^1, \mathrm{N}^1, \{K_i^1\}_{i=1}^{m}, \{\Diamond^{+\,1}_{k}\}_{k\in\mathbb{N}}, \mathrm{Today}^1, \top^0, \bot^0 \rangle.$$

Let us fix an enumerable set $Var := \{x_1, x_2, x_3, \dots\}$ of *propositional variables.* The formation rules for formulas over the propositional language $\mathcal{L}$ are given by the grammar:

$$\alpha ::= x_i \mid \alpha_1 \wedge \alpha_2 \mid \alpha_1 \vee \alpha_2 \mid \alpha_1 \to \alpha_2 \mid \neg\alpha \mid K_i\alpha \mid \mathrm{N}\,\alpha \mid \alpha_1\,\mathrm{Until}\,\alpha_2 \mid \Diamond_k^+\alpha \mid \top \mid \bot$$

Semantics of $\mathcal{TL}_{Dist}^M$. Our choice of the logical language for $\mathcal{TL}_{Dist}^M$ was motivated by the following semantic considerations.

For a binary relation $R \subseteq A \times A$ let

- $R^<$ be defined as follows: $aR^<b \iff aRb\,\&\,\neg(bRa)$;
- $R^2 = R \circ R$, $R^{n+1} = R \circ R^n$ — finite compositions of the relation R;
- $R^+ = \bigcup_{n=1}^{\infty} R^n$ — transitive closure of R;
- $R^* = \bigcup_{n=0}^{\infty} R^n$ — reflexive and transitive closure of R.

A *multi-agent cluster* is a Kripker frame $\langle C, R_1, \dots, R_m, R\rangle$, where 1) $R = C \times C$ is the universal relation on C; 2) $R_1, \dots, R_m$ are equivalence relations on C. From this point on, we will call multi-agent clusters simply *clusters*, since we will not consider any other type of them. The class of all clusters we denote by Cl. Given a cluster $C \in Cl$, we denote $R_{1,C}, \dots, R_{m,C}, R_C$ the respective relations.

A *chain* is a frame $\langle \bigcup_{i=1}^n C_i, R_1, \dots, R_m, R\rangle$, where $C_1, \dots, C_n \in Cl$ is a finite sequence of clusters, each R_j is the union of individual R_{j,C_i}'s, and

$$aRy \iff \exists i, j(i \leqslant j\,\&\,a \in C_i\,\&\,b \in C_j).$$

Let $\mathcal{C} = C(0), C(1), C(2), \dots$ be a countable sequence of clusters. The basic semantic objects upon which we define our logic are the Kripke models based on the following frames:

$$\mathcal{N}_{\mathcal{C}} := \left\langle \bigcup_{i\in Z} C(i) \cup \bigcup_{i\in Z} \lfloor C(i), C(i+1)\rfloor, R_1, \dots, R_m, R, \mathrm{Next} \right\rangle$$

where

1. for each $i \in \mathbb{N}$, $\lfloor C(i), C(i+1)\rfloor$ is a collection (may be infinite) of chains $\langle C_1, \dots, C_n\rangle$;
2. each R_j, $j = 1, \dots, m$ is the union of the respective $R_{j,C}$, i.e.,

$$R_j = \bigcup_{i\in\mathbb{N}} R_{j,C(i)} \cup \bigcup_{i\in\mathbb{N}} \{(R_{j,C} \mid C \in \lfloor C(i), C(i+1)\rfloor\}$$

3. $R = Q^+$, where

$$\begin{aligned} Q = {} & \textstyle\bigcup_{i\in\mathbb{N}} R_{C(i)} \cup \bigcup_{i\in\mathbb{N}}\{R_{Chain} \mid Chain \in \lfloor C(i), C(i+1)\rfloor\} \\ & \cup\{\langle a, b\rangle \mid a \in C(i)\,\&\,b \in C_1 \in \langle C_1, \dots, C_n\rangle \in \lfloor C(i), C(i+1)\rfloor\} \\ & \cup\{\langle a, b\rangle \mid a \in C_n \in \langle C_1, \dots, C_n\rangle \in \lfloor C(i), C(i+1)\rfloor\ \&\,b \in C(i+1)\} \end{aligned}$$

4. The relation Next is defined by

$$
\begin{aligned}
&a \,\mathrm{Next}\ b \iff \\
&\qquad (a \in C(i) \,\&\, b \in C(i+1)) \\
&\qquad \vee (a \in C \in \mathit{Chain} \in \lfloor C(i), C(i+1) \rfloor \,\&\, b \in C(i+1)).
\end{aligned}
$$

Computing truth-values of formulas. For any collection of propositional letters $\mathit{Prop} \subseteq \mathit{Var}$ and any frame $\mathcal{N}_{\mathcal{C}}$, a *valuation* in $\mathcal{N}_{\mathcal{C}}$ is a mapping, which assigns truth values to elements of Prop in $\mathcal{N}_{\mathcal{C}}$. Thus, for any $p \in \mathit{Prop}$, $V(p) \subseteq \mathcal{N}_{\mathcal{C}}$. We will call $\langle \mathcal{N}_C, V \rangle$ a (Kripke) model. For any such model $\mathcal{M}$, the truth values can be extended from propositions of Prop to arbitrary formulas. For $a \in \mathcal{N}_C$, we denote $(\mathcal{M}, a) \Vdash_V \phi$ to say that the formula ϕ is true at a in $\mathcal{M}_C$ w.r.t. valuation V.

In particular, $\forall p \in \mathit{Prop}$: $(\mathcal{M}, a) \Vdash_V p \iff a \in V(p)$ and boolean connectives are defined as usually.

For other operation, suppose $a, b \in \mathcal{N}_C$. Then

$$
\begin{aligned}
&(\mathcal{M}, a) \Vdash_V K_i\phi \iff \forall b(aR_ib \implies (\mathcal{M}, b) \Vdash_V \phi); \\
&(\mathcal{M}, a) \Vdash_V \mathrm{N}\,\phi \iff \forall b\,(a\,\mathrm{Next}\,b \implies (\mathcal{M}, b) \Vdash_V \phi); \\
&(\mathcal{M}, a) \Vdash_V \phi\,\mathrm{Until}\,\psi \\
&\iff \exists b\,(a\,\mathrm{Next}^*b \,\&\, (\mathcal{M}, b) \Vdash_V \psi \,\&\, \forall c(a\,\mathrm{Next}^*c\,\mathrm{Next}^+b \Longrightarrow (\mathcal{M}, c) \Vdash_V \phi); \\
&(\mathcal{M}, a) \Vdash_V \Diamond_k^+\phi \iff \exists b(a(R^<)^k b \,\&\, (\mathcal{M}, b) \Vdash_V \phi); \\
&(\mathcal{M}, a) \Vdash_V \mathrm{Today}\phi \iff \forall b \in C(a)\,((\mathcal{M}, b) \Vdash_V \phi).
\end{aligned}
$$

Informally, the connectives have the following meaning:

- $(\mathcal{M}, a) \Vdash_V K_i\phi$ — at the state a agent i knows that ϕ;
- $(\mathcal{M}, a) \Vdash_V \mathrm{Today}\,\phi$ — ϕ holds today (relatively to the time moment of a);
- $(\mathcal{M}, a) \Vdash_V \mathrm{N}\,\phi$ — ϕ holds tomorrow (counting from a);
- $(\mathcal{M}, a) \Vdash_V \Diamond_k^+\phi$ — ϕ holds in k steps from now (counting from a).

Note that: 1) connective $\Diamond_k^+$ is not a Kripke modality, in the sense, that there is no binary relation on a frame, which would define $\Diamond_k^+$ in the standard way (as in $a \Vdash \Diamond\phi \iff \exists b(aRb \,\&\, b \Vdash \phi)$; 2) although connective $\Diamond_0^+$, according to its definitions, is equivalent to Today, we prefer to retain name Today for clarity sake; 3) the connective Until cannot be defined from N as it is done, for instance, in *LTL*. It is rather a *weak* until.

Finally we arrive at definition:

Logic $\mathcal{TL}_{Dist}^M$ is the set of all formulas which are valid in all frames $\mathcal{N}_{\mathcal{C}}$.

Example: *Simulating software development cycles.*

Let us assume that clusters $C(i)$, $i \in \mathbb{N}$ correspond to stable releases of some software project, while clusters $C \in (C(i), C(i+1))$ correspond to "night-builds".

Each night-build C can potentially be graduated to the stable-release status (subject to its satisfying some specification requirements or passing some tests, etc., which is beyond considerations of this paper), which is indicated by the N-arrow from C to C_{i+1}. Each chain Ch of "night-builds" from the interval $(C(i), C(i+1))$ represents a possible track of software development, done by agents through the exchange of information, mediated by relations K_i (configuration of K_i's is specific to a particular multi-agent cluster). Although the "stable-releases" are not picked out by any operator in our language, they are distinguished semantically. In addition, according to the algorithm in Theorem 1, we can find out eventually what formulas hold or refuted in "stable-bulds".

3 Algorithm for Verification of Formula-Truths

For any logical system, one of most fundamental questions is the decidability problem: if there is an algorithm computing theorems of this logic. We address this problem to our logic $\mathcal{TL}^M_{Dist}$. The basic technique we use is based on the reduction of formulas in the language of $\mathcal{TL}^M_{Dist}$ to special inference rules and the verification of the validity these rules in frames $\mathcal{N}_{\mathcal{C}}$. This approach uses techniques to handle inference rules from [14,15,16,17].

Recall that an inference rule is a relation

$$r := \frac{\varphi_1(x_1,\dots,x_n),\dots,\varphi_l(x_1,\dots,x_n)}{\psi(x_1,\dots,x_n)},$$

where $\varphi_1(x_1,\dots,x_n),\dots,\varphi_l(x_1,\dots,x_n)$ and $\psi(x_1,\dots,x_n)$ are formulas constructed out of letters $x_1,\dots,x_n$. The letters $x_1,\dots,x_n$ are the variables of r, we use the notation $x_i \in Var(r)$.

A rule r is said to be *valid* in a Kripke model $\langle \mathcal{N}_C, V\rangle$ (notationally, $\mathcal{N}_C \Vdash_V r$) if

$$\forall a\ ((\mathcal{N}_C, a) \Vdash_V \bigwedge_{1\leq i\leq l} \varphi_i) \implies \forall a\ ((\mathcal{N}_C, a) \Vdash_V \psi).$$

Otherwise we say r is *refuted* in $\mathcal{N}_C$ (or *refuted in* $\mathcal{N}_{\mathcal{C}}$ *by* V), and write $\mathcal{N}_{\mathcal{C}} \nVdash_V r$. A rule r is *valid* in a frame $\mathcal{N}_C$ (notationally, $\mathcal{N}_C \Vdash r$) if, for any valuation V, $\mathcal{N}_C \Vdash_V r$.

Since our language $\mathcal{L}$ includes conjunction we can consider only rules with one-formula premise.

For any formula ϕ we can convert it into the rule $x \to x/\phi$ and employ a technique of reduced normal forms for inference rules as follows. Evidently,

Lemma 1. *A formula ϕ is a theorem of $\mathcal{TL}^M_{Dist}$ iff the rule $(x \to x/\phi)$ is valid in any frame $\mathcal{N}_{\mathcal{C}}$.*

A rule r_{rf} is said to be in *reduced normal form* if $r_{\mathrm{rf}} = \bigvee_{1\leq j\leq s} \theta_j/x_1$, where each θ_j has the form:

$$\theta_j = \bigwedge_{i=1}^{n} x_i^{t(j,i,0)} \wedge \bigwedge_{i=1}^{n}(\,\mathrm{Next}\, x_i)^{t(j,i,1)} \wedge \bigwedge_{i=1;l=1}^{n;m} (K_l x_i)^{t(j,l,i,1)}$$
$$\wedge \bigwedge_{i=1;l=0}^{n;k} (\Diamond_l^+ x_i)^{t(j,i,l,2)} \wedge \bigwedge_{i,l=1}^{n} (x_i \,\mathrm{Until}\, x_l)^{t(j,i,l,3)}$$

for some values $t(j,i,z), t(j,i,k,z) \in \{0,1\}$ and where, for every formula α above, $\alpha^0 := \neg\alpha$, $\alpha^1 := \alpha$.

Given a rule r_{rf} in the reduced normal form, r_{rf} is said to be a *normal reduced form for a rule* r iff, for any frame $\mathcal{N}_{\mathcal{C}}$,

$$\mathcal{N}_{\mathcal{C}} \Vdash r \iff \mathcal{N}_{\mathcal{C}} \Vdash r_{\mathrm{rf}}.$$

Using the technique similar to one described in [18, Section 3.1], we can transform every inference rule in the language $\mathcal{L}$ to a definably equivalent rule in the reduced normal form.

Lemma 2. *Every rule $r = \alpha/\beta$ can be transformed in exponential time to a definably equivalent rule r_{rf} in the reduced normal form.*

Proof. We shall specify for the language $\mathcal{L}$ the general algorithm described in [18, Lemma 3.1.3] and [18, Theorem 3.1.11].

Let $r = \alpha/\beta$ be an inference rule and suppose $\Diamond_0^+, \dots, \Diamond_k^+$ are all $\Diamond_i^+$'s that occur in r. We will need a set of new variables $Z = \{z_\gamma \mid \gamma \in Sub(r)\}$. Let us consider the rule in the *intermediate form*:

$$r_{\mathrm{if}} = z_\alpha \wedge \bigwedge_{\gamma \in Sub(r)\setminus Var(r)} (z_\gamma \leftrightarrow t(\gamma))/z_\beta,$$

where

$$t(\gamma) = \begin{cases} z_\delta * z_\epsilon & \text{, when } \gamma = \delta * \epsilon, \text{ where } * \in \{\wedge, \vee, \to, \mathrm{Until}\}, \\ *z_\delta & \text{, when } \gamma = *\delta, \text{ where } * \in \{K_1, \dots, K_m, \neg, \mathrm{N}\} \cup \{\Diamond_i^+\}_{i=0}^k, \\ \gamma & \text{, when } \gamma \in \{\bot, \top\}. \end{cases}$$

The rules r and r_{if} are equivalent. Indeed, suppose $\mathcal{M}$ is a model with a valuation $\nu : Var(r) \to 2^W$ over a frame $\mathcal{N}_{\mathcal{C}}$, such that $\mathcal{M} \nVdash r$. It means that $\mathcal{F} \Vdash_\nu \alpha(\bar{x})$, and there exists $w \in \mathcal{F}$ such that $(\mathcal{F}, w) \nVdash_\nu \beta(\bar{x})$ Let $\mu : Z \to 2^W$ be the valuation defined as follows: $\mu(z_\gamma) = \nu(\gamma)$. It is straightforward to show that $\mathcal{F} \Vdash_\mu z_\alpha \wedge \bigwedge_{\gamma\in Sub(r)\setminus Var(r)}(z_\gamma \leftrightarrow t(\gamma))$. In addition, $(\mathcal{F}, w) \nVdash_\mu z_\beta$.

For the other direction, suppose $\mathcal{F} \Vdash_\mu z_\alpha \wedge \bigwedge\{z_\gamma \leftrightarrow t(\gamma) \mid \gamma \in Sub(r)\setminus Var(r)\}$ and $(\mathcal{F}, w) \nVdash_\mu z_\beta$, for some valuation $\mu : Z \to 2^W$ and some $w \in W$. Define $\nu : Var(r) \to 2^W$ by $\nu(x_i) = \mu(z_{x_i})$. It follows directly that for all $\gamma \in Sub(r)$, $\nu(\gamma) = \mu(z_\gamma)$. Thus $\mathcal{F} \Vdash_\nu \alpha(\bar{x})$, $(\mathcal{F}, w) \nVdash_\nu \beta(\bar{x})$, hence $\mathcal{F} \nVdash_\nu r$.

Finally, we transform the premise of the obtained rule r_{if} into a perfect disjunctive normal form over primitives of the form x_i, $\mathrm{N}\, x_i$, $K_j x_i$, $\diamond_j^+ x_i$ and $x_i \,\mathrm{Until}\, x_l$. This requires no more than exponential time on the number of variables, i.e., on the number of subformulas of the original rule (the same as for reduction of any boolean formula to the perfect disjunctive normal form).

Decidability of $\mathcal{TL}_{Dist}^{M}$ will follow (by Lemma 1) if we find an algorithm recognizing rules in the reduced normal form which are valid in all frames $\mathcal{N}_C$. We need one more construction on Kripke frames, which we describe below.

Suppose we have a frame $\mathcal{F}$ which looks like a finite initial fragment of some $\mathcal{N}_C$, the end cluster of which is attached, by extending N and R-relations, to some previous cluster in the segment (provided $\mathcal{N}_C$ is given, this construction can be parameterized by two natural numbers). We will call such frames *of* $\mathcal{N}_C^{\sharp}$ *type*. For any valuation V of letters from a formula ϕ in $\mathcal{N}_C^{\sharp}$ the truth value of ϕ can be defined on elements of $\mathcal{N}_C^{\sharp}$ by the rules similar to ones given for the frames $\mathcal{N}_C$ above (just in accordance with the meaning of logical operations). Due to size limitations we omit a detailed description of these rules.

Lemma 3. *A rule* r_{rf} *in the reduced normal form is refuted in a frame* $\mathcal{N}_C$ *iff* r_{rf} *can be refuted in some frame* $\mathcal{N}_C^{\sharp}$ *by a valuation* V *of special kind, where the size of the frame* $\mathcal{N}_C^{\sharp}$ *is triple exponential in* r_{rf}*.*

From Theorem 2, Lemma 1 and Lemma 3 we derive

Theorem 1. *The logic* $\mathcal{TL}_{Dist}^{M}$ *is decidable. The algorithm for checking a formula to be a theorem of logic* $\mathcal{TL}_{Dist}^{M}$ *consists in verification of validity rules in the reduced normal form at frames* $\mathcal{N}_C^{\sharp}$ *of size triple-exponential in the size of reduced normal forms with respect to valuations of special kind.*

4 Conclusions and Future Work

This paper continues the work of the authors aiming to find an adequate formal language for describing interactions in multi-agent systems. The main goal and difficulty is to strike right balance between expressiveness of the language and its good meta-logical properties, like those that are demonstrated by some traditional "philosophical" logics. Given such balance, the next big goal is to be able to derive "protocols" or, at least, general principles of agents' interactions, that ensure good properties of the system, described by some specification.

From a mathematical point of view, this paper develops a technique for constructing mathematical models for modeling transitions between discreet time moments in linear temporal frames with agents and distance operations. In particular, we define the multi-modal logic $\mathcal{TL}_{Dist}^{M}$, which describes such models. Technically our main aim is to find an algorithm, which can compute logical laws of $\mathcal{TL}_{Dist}^{M}$. The problem is resolved via model theoretic constructions on special Kripke models and use of inference rules in the reduced form. In particular, it solves the satisfiability problem for $\mathcal{TL}_{Dist}^{M}$.

The major open issues for this construction are: the question of finite axiomatization for $\mathcal{TL}_{Dist}^{M}$, complexity issues and possible ways of refining the complexity bounds in the algorithm.

References

1. Pnueli, A.: The temporal logic of programs. In: Proc. of the 18th Annual Symp. on Foundations of Computer Science, pp. 46–57. IEEE, Los Alamitos (1977)
2. Manna, Z., Pnueli, A.: Temporal Verification of Reactive Systems: Safety. Springer, Heidelberg (1995)
3. Barringer, H., Fisher, M., Gabbay, D., Gough, G.: Advances in Temporal Logic. Applied logic series, vol. 16. Kluwer Academic Publishers, Dordrecht (1999)
4. Vardi, M.Y.: Reasoning about the past with two-way automata. In: Larsen, K.G., Skyum, S., Winskel, G. (eds.) ICALP 1998. LNCS, vol. 1443, pp. 628–641. Springer, Heidelberg (1998)
5. van Benthem, J.: The Logic of Time. Kluwer, Dordrecht (1991)
6. van Benthem, J., Bergstra, J.: Logic of transition systems. Journal of Logic, Language and Information 3(4), 247–283 (1994)
7. Gabbay, D., Hodkinson, I.: An axiomatisation of the temporal logic with until and since over the real numbers. Journal of Logic and Computation 1(2), 229–260 (1990)
8. Hodkinson, I.: Temporal logic and automata, chapter ii of temporal logic. In: Gabbay, D.M., Reynolds, M.A., Finger, M. (eds.) Mathematical Foundations and Computational Aspects, vol. 2, pp. 30–72. Clarendon Press, Oxford (2000)
9. de Jongh, D., Veltman, F., Verbrugge, R.: Completeness by construction for tense logics of linear time. In: Troelstra, A., Visser, A., van Benthem, J., Veltman, F. (eds.) Liber Amicorum for Dick de Jongh. Institute of Logic, Language and Computation, Amsterdam (2004)
10. Kacprzak, M.: Undecidability of a multi-agent logic. Fundamenta Informaticae 45(2-3), 213–220 (2003)
11. Fagin, R., Halpern, J., Moses, Y., Vardi, M.: Reasoning About Knowledge. MIT Press, Cambridge (1995)
12. Rybakov, V.: Logic of discovery in uncertain situations– deciding algorithms. In: Apolloni, B., Howlett, R.J., Jain, L. (eds.) KES 2007, Part II. LNCS (LNAI), vol. 4693, pp. 950–958. Springer, Heidelberg (2007)
13. Rybakov, V.: Multi-agent logics with interacting agents based on linear temporal logic: Deciding algorithms. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 1243–1253. Springer, Heidelberg (2008)
14. Rybakov, V.: Rules of inference with parameters for intuitionistic logic. Journal of Symbolic Logic 57(3), 912–923 (1992)
15. Rybakov, V.: Linear temporal logic with until and next, logical consecutions. Annals of Pure and Applied Logic 155(1), 32–45 (2008)
16. Rybakov, V.: Logical consecutions in discrete linear temporal logic. Journal of Symbolic Logic 70(4), 1137–1149 (2005)
17. Rybakov, V.: Logical consecutions in intransitive temporal linear logic of finite intervals. Journal of Logic and Computation 15(5), 663–678 (2005)
18. Rybakov, V.: Admissible Logical Inference Rules. Studies in Logic and the Foundations of Mathematics, vol. 136. Elsevier Sci. Publ., North-Holland, Amsterdam (1997)

On Data Representation in Reactive Systems Based on Activity Trace Concept

Krzysztof Skrzypczyk

Silesian University of Technology, Akademicka 16, 44-100 Gliwice, Poland
krzysztof.skrzypczyk@polsl.pl

Abstract. The paper addresses the problem of data representation collected during the work of a robot controlled by reactive, behaviour-based system. This type of systems are intended to work with low accuracy sensors. That implies there are many problems with building models of a workspace. In this work some aspects of processing, storing and using this data are presented. The usage of the concept named activity trace is discussed in the work.

1 Introduction

The autonomy of a mobile robot is understood as an ability of the robot to operate without supervision of a human operator. In order to face this requirement the control system must handle and solve a lot of complex problems. The process of building the world representation is based on the data provided by sensory subsystem. It have to be emphasised that sensors provide data that is often inaccurate and uncertain therefore classical, planner-based, deliberative approaches to the mobile robot control system design often fail. Therefore more efficient and faster methods of the mobile robot collision free movement control were developed. One of them is a purely reactive architecture introduced in [4] which implements a control strategy as a collection of stimulus-reaction pairs. Behaviour based control system architecture [2][5] embodies some of the properties of reactive systems and may contain reactive components. One of the key issues that appears while designing the behaviour based systems is just the representation of the knowledge about the environment [5][6][7][8][9]. Reactive systems work on the base of temporal, local information continuously processing input stimuli into reaction. In such case, if the workspace configuration is complex the system might not be able to complete the navigational task. Without representation of the past activity functionality of these systems is limited. The idea of the proposed approach, named activity trace, is to store in a time ordered way the knowledge of events that happened during an operation of the robot in the past. What is crucial for this method is the fact that these events are marked and recognised by distinctive sequences of the behaviours activity. The first application of the method is navigational dead-locks detection that may occur during the robot's navigation. The second one is a map building process. The advantage of the proposed form of representation is that it does not

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 345–352, 2010.
© Springer-Verlag Berlin Heidelberg 2010

consume much memory resources and is easy to process. In the paper examples of application of this method are presented. Simulation experiments proved the proposed approach to be effective.

1.1 The Robot and Its Sensory System

Hereafter the control of a differentially driven disc-shaped mobile robot will be considered. Let us assume that the robot is equipped with low cost, low accuracy and uncertain sensory system. The pose of the robot is denoted by a vector:

$$R = [x_R, y_R, \theta_R]^T \tag{1}$$

where x_R, y_R denote coordinates of the centre of the robot and θ_R is the heading of the robot. The control of the robot is given by:

$$U = [\omega, v]^T \tag{2}$$

where (ω, v) are the angular and the linear velocity of the robot. Sensors the robot is equipped with are numbered form 0 to 7 and they are mounted on the circumference of the robot body. Their readings are denoted by the vector:

$$S = [s_0, s_1, ..., s_7]^T,\ s_i \in \langle 0, 1023 \rangle \tag{3}$$

In order to reduce an amount of processed information virtual sensors s_L, s_F, s_R are defined that collect data from the real sensors placed on the left, front and right side of the robot correspondingly. The fixing points of the sensors are denoted by:

$$M_S = [m_{s,0}, m_{s,1}, ..., m_{s,7}]^T,\ m_{s,i} = \{x_{s,i}, y_{s,i}\} \tag{4}$$

Geometrical location of virtual sensors is calculated as the mean value of the real sensors and denoted by m_{SL}, m_{SF}, m_{SR}.

1.2 Control System Framework

Design of the control system used in this work is based on behaviour-based idea of control [2][4]. The system is composed of behaviours that process the robot's state and sensory information into the set-points (2)for the motion controller. A general diagram of the system is presented in fig.1. It is easy to distinguish following modules of the system: *behaviour definition module, arbitration module, control computation module, task execution manager, dead lock detector, mapping module.*

In the discussed system eight behaviours were implemented. First four of them (*AVOID LEFT*, *AVOID FRONT*, *AVOID RIGHT*, *SPEED UP*) are responsible for avoiding collisions with objects. Fifth behaviour (*TRACK GOAL*) minimises the distance between the robot and the target. Behaviour *STOP* simply stops the robot in case a collision is detected or the target is reached. Sixth behaviour named *STROLL* makes the robot goes straight in case when no object is detected. And the last behaviour - *PASS NARROW* stabilises robot's movement preventing oscillations during going through narrow passages. In this work the method of priority arbitration was used. The activation level of the selected kth behaviour constitutes the basis for the robot control computation. The control vector (2) is defined by heuristic function of activation level of the selected behaviour $U = f_k(a_k)$.

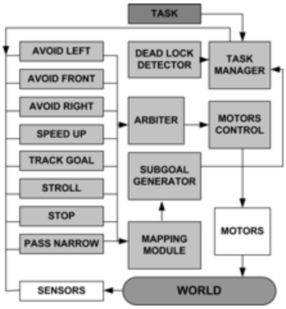


Fig. 1. General diagram of the control system

The purpose of the task execution manager module is to supervise the execution of the elementary navigational task. More detailed description of the work of this module can be found in [6].

2 Behaviour Activity Trace

The key issue of the proposed approach is that the system does not store and analyses information abut the shape of the environmental objects but it utilises information about events. Since the events are caused by a configuration of environmental objects, The place the robot was located in time the given behaviour was activated can be perceived as a part of a symbolic map. These spots are named hereafter *distinctive points.*

Definition 1. *The distinctive point CP_k is the point $(x_{CP,k}, y_{CP,k})$ defined in the cartesian space determined by the location of the center of the robot (x_r, y_r) recorded in a moment when* k*th distinctive event occurred.*

For purpose of this work seven, selected distinctive events were defined. Occurrence of each distinctive event causes the appropriate distinctive point $CP_i \quad i \in [1, 7]$ is registered. Detailed description of the distinctive events can be found in [6].

Definition 2. *The activity trace is the time ordered sequence of distinctive points:*

$$T = \left\{CP_k^0, CP_k^1, ..., CP_k^{n-1}, CP_k^n\right\} \quad k = 1, 2, ..., 7 \tag{5}$$

As can be seen the activity trace is the record of past activity of the robot by means of distinctive events. The events have been recorded since the moment of the beginning of the navigation process t=0 till the present $t = n\Delta t$ time. Analysing behaviour activity traces some useful information about the structure of the workspace can be get out.

3 Dead Lock Detection

The dead lock, in the meaning of navigational process, can be defined as a situation the robot is not able to continue the movement toward the target of navigation. Such occurrence manifests in repeating by the robot some periodic sequence of action that results in keeping the robot inside of some region. This phenomenon is often addressed as local minima problem. On the base of multiple simulations it was stated that the situation when the robot is not able to reach the target is mainly caused by specific object's layout that form u-shaped lay-by located on the course of the vehicle. Such situation can be easily detected using behaviour activity trace concept. It was noticed that dead end situation described above corresponds to an occurrence of three element chunk of the activity trace CP_2, CP_3, CP_2. Moreover a mutual location of the distinctive points is checked. If all of them are inside of a circle of a radius r_T that means the navigation algorithm failed. Detected dead-lock location is recorded in a buffer and denoted by coordinates $x_{d,i}, y_{d,i}$ where $i = 1, 2, ..., N_d$. The quantity N_d denotes the number of all detected dead locks. In such case the recovery algorithm should be turned on.

4 The Map Building Method

4.1 The Grid

In order to build the environmental map the rectangular grid is put on the workspace area. Since the robot body is assumed to be the compact one it can be approximated by a circle the size of the grid cell ΔG is equal to a diameter of the circle. That causes the grid is not the dense one. Of course small resolution of the grid results in low precision of the map. Let us denote the grid as:

$$G = \{g_{i,j}\} \;\; i = 1, 2, ..., N, \;\; j = 1, 2, ..., M \tag{6}$$

4.2 The Map Structure

The map can be perceived as a source of information about two features of the workspace: occupancy of the world by an obstacle, and information the given part of the workspace is free. The map based on the grid has a form of weighted graph. Each vertex of the graph corresponds to the given cell. Information about both geometrical and symbolical features of the cell is related to the given vertex. The edges of the graph represent costs of transition between the cells of the map.

The higher belief the given cell is not occupied the lower is the cost of transition to the given cell. Let us denote the map as the graph $M_G = (G, E)$ where G denotes the set of vertices defined by (6). Each vertex $\mathbf{g_{i,j}}$ is described by the vector of the three features:

$$\mathbf{g_{i,j}} = \left[C_{G,i,j}, P^F_{i,j}, P^O_{i,j}\right]^T \tag{7}$$

where $C_{G,i,j} = (x_{C,i,j}, x_{C,i,j})$ denotes the geometrical center of the specified cell. The $P^F_{i,j} \in \langle 0,1\rangle$ is the factor that reflects the degree of belief that the cell $g_{i,j}$ is free. Similarly the $P^O_{i,j} \in \langle 0,1\rangle$ is the factor that represents the certainty that the given cell is occupied by an obstacle. Second element of the map definition E is the set of edges of the graph. The edges represent the possibilities of transitions between individual vertices. They are defined by:

$$E \subseteq \left\{g_{i,j}, g_{k,l}, w^{k,l}_{i,j} : g_{i,j}, g_{k,l} \in G, w^{k,l}_{i,j} \in \Re\right\} \tag{8}$$

where $w^{k,l}_{i,j}$ $i, k = 1, 2, ..., N,$ $j, l = 1, 2, ..., M$ is the weighting factor related to the given transition.

4.3 The Map Acquisition

While the robot is navigating toward the goal the behaviours activities are recorded. When the robot goes through the empty area the behaviour *STROLL* is activated. In this case a sequence of distinctive points can be distinguished in the activity trace T. In order to avoid uncertainty caused by some interferences a sequence of length B is taken into account. If the aforementioned sequence is detected in discrete moment of time t_n the robot location inside of the grid is determined:

$$R_G(t_n) = (q, r),\ q = 1...N, r = 1...M \tag{9}$$

Then for all neighbouring cells $(g_{i,j} : i = p-1, p, p+1, j = \ r-1, r, r+1)$ factors $P^F_{i,j}$ are computed according to the following equation:

$$P^F_{i,j}(t_n) = 1 - \frac{\sqrt{2}}{2\Delta G} d^R_{i,j}(t_n) \tag{10}$$

where $d^R_{i,j}$ denotes the distance between the robot and the given cell. The ΔG in (10) is the size of the cell. The robot can visit the same place in the map many times. Therefore a procedure of the map updating is needed. Let us denote as $P^{F,OLD}_{i,j} = P^F_{i,j}(t_p < t_n)$ the value recorded in the map in the past time. The new value saved to the map is determined as:

$$P^F_{i,j} = \begin{cases} P^F_{i,j} \quad \text{for } P^F_{i,j} > P^{F,OLD}_{i,j} \wedge P^O_{i,j} = 0 \\ P^{F,OLD}_{i,j} \text{ for } P^F_{i,j} \leq P^{F,OLD}_{i,j} \wedge P^O_{i,j} = 0 \\ 1 - P^O_{i,j} \text{ for } P^O_{i,j} > 0 \end{cases} \tag{11}$$

Using this procedure while the robot navigates toward the target the map of the free space is created. It will be used further as a cue for navigation inside of the partially known workspace.

Another problem is to determine the value of the factor $P^O_{i,j}$ that expresses the belief that the given cell is occupied by an object. In order to do it we use the activity trace of behaviours that are responsible for collision avoidance. Since the value of the activation level generated by each of these behaviours depends strictly on a proximity to the obstacle it can be used for computing the value of the factor $P^O_{i,j}$ (7). Therefore the presence of one of the sequences $\left\{CP^1_k, ..., CP^B_k\right\}$ $k = 4, 5, 6$ in (5) must be detected. If it is detected the cell that is thought to be occupied by an obstacle must be distinguished. Let us denote the number *(i,j)* of the cell in the moment t_n when one of the aforementioned events was detected:

$$(i,j) = \begin{cases} (i_4, j_4) \text{ if } m_{SL} \text{ lies inside of } g_{i_4,j_4} \\ (i_5, j_5) \text{ if } m_{SF} \text{ lies inside of } g_{i_5,j_5} \\ (i_6, j_6) \text{ if } m_{SR} \text{ lies inside of } g_{i_6,j_6} \end{cases} \tag{12}$$

The value of the factor $P^O_{i,j}$ can be determined as the activity level of the appropriate behaviour - related to the given event:

$$P^O_{i,j} = a_k, \; k = 4, 5, 6 \tag{13}$$

where a_k denotes the value of the activity of the behaviour *AVOID LEFT*, *AVOID FRONT*, *AVOID RIGHT* correspondingly. Finally using previous calculations the weighting factors can be determined:

$$w^{k,l}_{i,j} = \frac{1}{P^F_{k,l}} d^{k,l}_{i,j} \quad \text{or} \quad w^{k,l}_{i,j} = \frac{1}{1 - P^O_{k,l}} d^{k,l}_{i,j} \tag{14}$$

The weighting factor is calculated with accordance to the (14) depending on the data stored in the $g_{k,l}$ vertex. The $d^{k,l}_{i,j}$ in 14 is the euclidean distance between the center points $C_{G,i,j}$ and $C_{G,k,l}$.

5 Simulation

In order to verify the proposed approach a number of simulations were carried out. The result of the simulation without any information about the workspace structure is presented in fig 2a. The path of the robot resulted has a typical reactive nature. The robot is driven straight toward the target and its path is disturbed by surrounding obstacles. While the robot is executing the task, data about the structure of the environment is collected. The map acquired during the task is shown in fig. 2b. The values of the factors $P^F_{i,j}$ and $P^O_{i,j}$ are illustrated using gray scale. In case of the $P^F_{i,j}$ the black corresponds to 0 and the white to 1. The factor $P^O_{i,j}$ is illustrated the other way round. The dark colours denote the regions that are recognised as occupied by obstacles or uncertain. The traverse able part of the recognised environment is marked with light colours. In order to show that

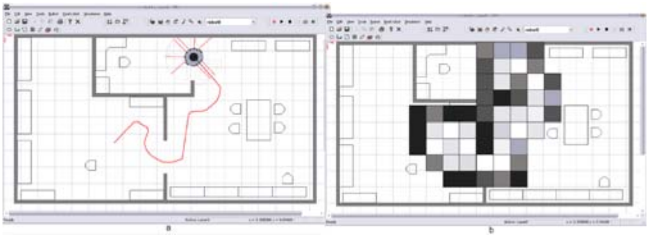


Fig. 2. The result of the navigational experiment (a) and the acquired map (b)

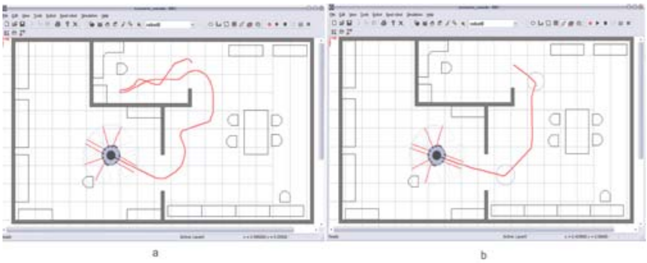


Fig. 3. The result of the reverse navigational experiment without the use of the map (a). The same experiment carried out using acquired map (b).

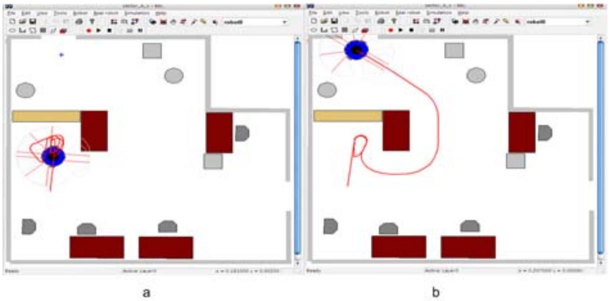


Fig. 4. The result of the simulation with dead lock detection(a) and the recovery algorithm action (b)

a reverse experiment was arranged. First the experiment was performed without use of the previously acquired information about environment. The result of simulation made without using the map acquired is shown in fig. 3a. It is easy to see that the task was not executed effectively. In the next experiment the system utilised the knowledge acquired. Using information the map contains the sub-goal generator module determined 2 sub-goals that lie on the shortest path between the start point and the goal. As can be seen in fig 3b the task was performed more effectively. The sub-goals are marked with circles. Figure 4a presents the result of the simulation of the system without using the dead lock detector and recovery algorithm. It is easy to see that the robot stuck in the dead lock inside of an u-shaped obstacle. In fig. 4b the result of a work of the dead lock detector and recovery algorithm is shown. The algorithm got the robot out of the dead lock.

6 Conclusion

In this paper the novel method of data representation in behaviour based systems was presented. The process of environmental maps building based on behaviour activity was described in this work. Multiple simulations proven the method work well and the proposed model of data representation can improve navigation process. It must be stressed again that the advantage of the method is its simplicity. There is no need to do complex computations in terms of probabilistic. Processing and updating the model is intuitive and can be done using simple heuristics.

Acknowledgements. This work has been supported by Ministry of Science and Higher Education funds in the years 2009 - 2010 as development project OR00013409.

References

1. Althaus, P., Christensen, H.I.: Behaviour coordination in structured environments. Advanced Robotics 17(7) (2003)
2. Arkin, R.C.: Behavior-Based Robotics. MIT Press, Cambridge (1998)
3. Bicho, E., Schoner, G.: The dynamic approach to autonomous robotics demonstrated on a low-level vehicle platform. Robotics and Autonomous Systems 21(1) (1997)
4. Brooks, R.A.: Intelligence without representation. Artificial Intelligence (47) (1991)
5. Michaud, M., Mataric, M.J.: Learning from history for behavior-based mobile robots in nonstationary Conditions. Special issue on Learning in Autonomous robots, Machine Learning 31(1-3) (1998); Autonomous Robots 5(3-4) (1998)
6. Skrzypczyk, K.: Behavior activity trace method. Application to dead lock detection in a mobile robot navigation. In: Proc. of 4th International Conference on Informatics in Control, Automation and Robotics, Robotics and Automation, Angers, France, vol. 2, pp. 265–269
7. Fabrizia, E., Saffiotti, A.: Augmenting topology-based maps with geometric information. Robotics and Autonomous Systems 40(2-3), 91–97
8. Mataric, M.J.: Learning in behavior-based multi-robot systems: policies, models and other agents. Cognitive Systems Research 2(1), 81–93 (2001)
9. Wang, F.: Knowledge Representation in a Behavior-Based Natural Language Interface for Human-Robot Communication. In: Huang, D.-S., Li, K., Irwin, G.W. (eds.) ICIC 2006. LNCS (LNAI), vol. 4114, pp. 730–735. Springer, Heidelberg (2006)

Part IV

Various Problems of Artificial Intelligence

Optimization of the Height of Height-Adjustable Luminaire for Intelligent Lighting System

Masatoshi Akita[1], Mitsunori Miki[2],
Tomoyuki Hiroyasu[3], and Masato Yoshimi[2]

[1] Graduate School of Engineering, Doshisha Univ.
[2] Department of Science and Engineering, Doshisha Univ.
[3] Department of Life and Medical Sciences, Doshisha Univ.
1-3 Tatara Miyakodani Kyotanabe-shi, Kyoto, Japan, 610-0321
{makita@mikilab,mmiki@mail,tomo@is,myoshimi@mikilab}.doshisha.ac.jp

Abstract. We are doing reseaching and development of an intelligent lighting system to provide desired brightness to a desired place. In this study, we consider how to satisfy target illuminance for each worker and minimize power consumption with not only luminous intensity for each light but also the height of luminaire as the design variables. Therefore, highly-accurate target illuminance as well as very high illuminance can be realized and further energy saving is achieved.

Keywords: lighting control, system, optimization, intellectual productivity, energy saving.

1 Introduction

In recent years, intelligence has been incorporated in various systems including electric appliances and automobiles by autonomously controlling systems' own movements depending on the user or environment, leading to reduction of burdens on people. Intelligence relates to artifacts to make judgments based on their own knowledge obtained from sensors, etc. and take proper movements.

Intelligence for lighting systems was extremely low, although various systems have been acquiring intelligence. With common lighting systems in these days, the illumination pattern relies on power layout and switches, and such illumination pattern desired by a user may not be achieved in some cases. In addition, excessive brightness tends to be provided since there is no function to change the brightness of lights.

Under these circumstances, we are studying the intelligent lighting system to realize improvement of intellectual productivity as well as reduction of power consumption by providing brightness required individually (illuminance). As a result of basic experiments, this system is found to satisfy illuminance required by a worker and also realize high energy savings, and verification experiments are carried forward in the actual environment for practical application.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 355–362, 2010.
© Springer-Verlag Berlin Heidelberg 2010

Since luminaires have a wide radiation angle in the case of current intelligent lighting systems, distribution of required illuminance might not be physically achieved in some cases, including the case that workers require significantly different levels of illuminance from each other or the case that illuminance beyond the capacity of luminaires is required. Therefore, a method to simultaneously satisfy various illuminance levels required by workers is proposed in this study by using height-adjustable luminaires, with not only luminous intensity for each light but also the luminaire height as the design variables.

2 Intelligent Lighting System

2.1 Components of Intelligent Lighting System

With the intelligent lighting system, multiple luminaires independently adjust the brightness of the lights (luminous intensity) to realize illuminance required by workers[1]D

In the intelligent lighting system, multiple dimmable lights, the light control device attached, multiple movable illuminance sensors, and the electric meter are connected to one network. Since a distributed autonomous optimization algorithm is incorporated in the control device installed in each lighting equipment, it can operate as the distributed autonomous system as a whole. The composition of the intelligent lighting system is indicated in Fig.1.

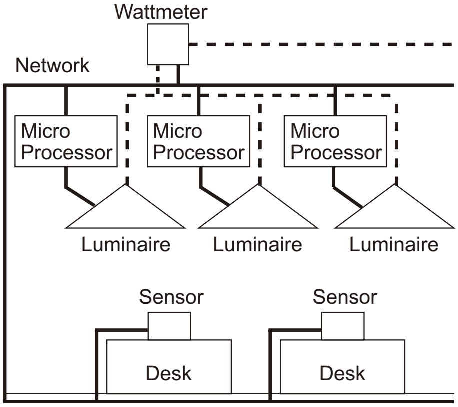


Fig. 1. Composition of the intelligent lighting system

2.2 Control Algorithm for the Intelligent Lighting System

In the intelligent lighting system, each worker enters desired illuminance (target illuminance) in the system. The microprocessor built in each light controls its own luminous intensity by using the distributed autonomous algorithm. Adaptive Neighborhood Algorithm using Correlation Coefficient (ANA/CC) [4] is used as the optimization method.

ANA/CC is the improvement to control lights based on Simulated Annealing (SA). In SA, the subsequent solution is generated randomly around the current solution, and accepted based on the change in the objective function value as well as on the temperature parameter. It is the algorithm to obtain the (local) optimal solution by repeating this. In ANA/CC, the variation range of the design variable is changed instead of the temperature parameter, and the schematic mutual positions are kinetically learned in accordance with the correlation between the light and illuminance sensor. Based on the correlation, the design variable, i.e., the range to generate the next luminous intensity is adaptively determined. In ANA/CC, the correlation coefficient is calculated based on the change in luminous intensity as well as in illuminance for each light. The control flow for ANA/CC is described in the following:

1. Establish initial parameters including the initial luminous intensity.
2. Illuminate each light at the initial luminous intensity.
3. Obtain sensor information for each illuminance sensor (sensor ID, current illuminance and target illuminance) and power usage measured with an electric meter, based on which the objective function value is calculated.
4. Determine the proper range to generate the next luminous intensity based on the correlation coefficient.
5. Randomly generate the next luminous intensity within the range determined in Step 4 and illuminate the light at the luminous intensity.
6. Obtain sensor information for each illuminance sensor as well as the power usage with the electric meter again.
7. Calculate the correlation coefficient from the current illuminance obtained and the luminous intensity changed.
8. Calculate the objective function value under the illuminated condition with the changed luminous intensity, based on the sensor information and power usage.
9. If the objective function value turns good, confirm the luminous intensity and return to Step 3.
10. If the objective function value turns poor in Step 8, cancel the changed luminous intensity and return to Step 3.

Based on the above movement, the mutual location relationship between the light and illuminance sensor is understood using the correlation coefficient, to achieve the target illuminance with unwasted movement and at the same time promptly converge into the energy-saving mode. The reason to return to Step 3 rather than Step 4 in Steps 9 and 10 is to respond to kinetic changes in the environment such as shift of illuminance sensor and incoming radiation of outside light.

2.3 Achievement of Target Illuminance

It has already been confirmed that the intelligent lighting system is able to provide illuminance required by each worker and realize energy saving, while there are possible situations where distribution of desired illuminance cannot be physically achieved in some cases. For example, they include a case that adjacent workers require illuminance levels significantly different from each other, a case that some workers require very high illuminance, and a case that a worker requiring high illuminance is in the corner of a room.

In order to satisfy these types of lighting requirements which are difficult to achieve, it is necessary to make the height of luminaires adjustable or to change workers' seating layout. In this study, distribution of illuminance levels that could not have been realized in the past is realized, by making the height of luminaires adjustable vertically to the ceiling surface (height-adjustable luminaires) the design variable.

3 Optimization of the Luminaire Height to Achieve Personal Illuminance

3.1 Control Algorithm

As a method to satisfy the requirement mentioned in Section 2.3, we propose a method to achieve required illuminance that could not have been realized in the past with minimum energy, by using height-adjustable luminaires and making the height of each luminaire the design variable. The control algorithm is indicated in the following:

1. Fix the height of luminaires, optimize the luminous intensity for each light by using ANA/CC, and obtain the objective function value under this condition.
2. Randomly change the height of all luminaires.
3. Optimize the luminous intensity for each light under the condition that the height of luminaires is changed.
4. Obtain the objective function value again. If the value turns good, maintain the present height of luminaires. If the value turns poor, return to the height of luminaires before moving.
5. Repeat the operation in Steps 2 to 4, and end it when illuminance required by all workers is achieved or when processing is performed more than a certain number of times.

The objective function used in this algorithm is indicated in Eq.(1):

$$f = P + w\sum_{i=1}^{n} g_i \tag{1}$$

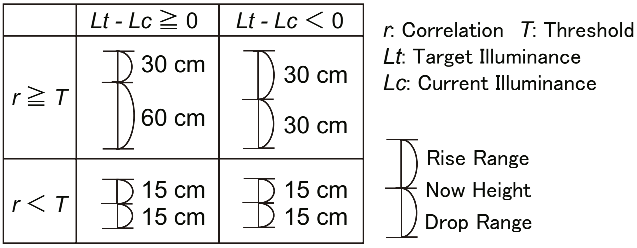
$$g_i = \begin{cases} (Lt_i - Lc_i)^2 & 50 \leq |Lt_i - Lc_i| \\ 0 & otherwise \end{cases}$$

P: power consumption, w: weight, Lc: current illuminance, Lt: target illuminance, n: number of illuminance sensors

The objective function consists of power consumption P and illuminance difference g_i, and priority of power consumption reduction over illuminance achievement is determined by changing the weight w. g_i is the value to be added when the difference between the target illuminance and the current illuminance is 50 lx or more, and the square of the difference is used.

3.2 Change of Luminaire Height

A method to determine the variation range of the height in regards to height-adjustable luminaires is discussed. As mentioned in Section 2.2, the location relationship between a light and illuminance sensor is schematically understood by obtaining the correlation coefficient based on the change in luminous intensity and illuminance for each light in ANA/CC. By setting the condition to the randomly changing height of luminaires in accordance with this schematic location relationship, the target illuminance seems to be satisfied with less number of search times, quickly converging into a state of minimum energy. The one-time variation range of the luminaire height is indicated in Fig.2.



	Lt - $Lc \geqq 0$	Lt - $Lc < 0$
$r \geqq T$	30 cm 60 cm	30 cm 30 cm
$r < T$	15 cm 15 cm	15 cm 15 cm

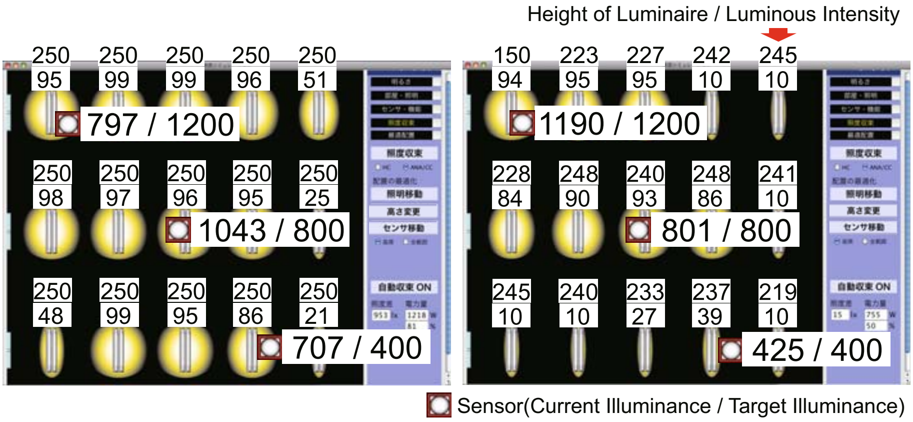
Fig. 2. One-time variation range of luminaire height

As indicated in Fig.2, illuminance is deficient even though the intended light influences the illuminance sensor, when the correlation coefficient between a light and illuminance sensor is more than the threshold and the current illuminance is lower than the target illuminance. In this case, higher illuminance is achieved by reducing the luminaire height. For this purpose, the downside of the variation range of the luminaire height is prioritized. On the contrary, the luminaire height is changed to even out the top and bottom, when the correlation coefficient is more than the threshold and the current illuminance is higher than the target illuminance. When the correlation coefficient is smaller than the threshold, the influence of the light is considered to be small; therefore the variation range of the luminaire height is narrowed to eliminate wasteful searching without relying on the values of current illuminance and target illuminance. By designing the one-time variation range of the luminaire height in this way, the luminaire becomes low in a place requiring illuminance and the illuminance required by a worker is achieved.

3.3 Evaluation Experiment of the System

When extremely high illuminance levels are required. The evaluation experiment is conducted by simulation in order to verify the effectiveness of the algorithm proposed. The criterion to determine convergence of illuminance is when the difference between the target illuminance and the current illuminance is within 50 lx. This is because people are not able to recognize the difference in illuminance at 50 lx level[5]D

First of all, the verification experiment is conducted in the case that some workers require extremely high levels of illuminance. In this experiment, a situation where one out of three workers requires very high illuminance under the environment of 15 white fluorescent lamps is assumed. The results when only the luminous intensity for each light is optimized without changing the luminaire height are indicated in Fig.3-(a), and the results when the luminaire height is optimized from that situation are indicated in Fig.3-(b).



(a) Optimization of the luminance

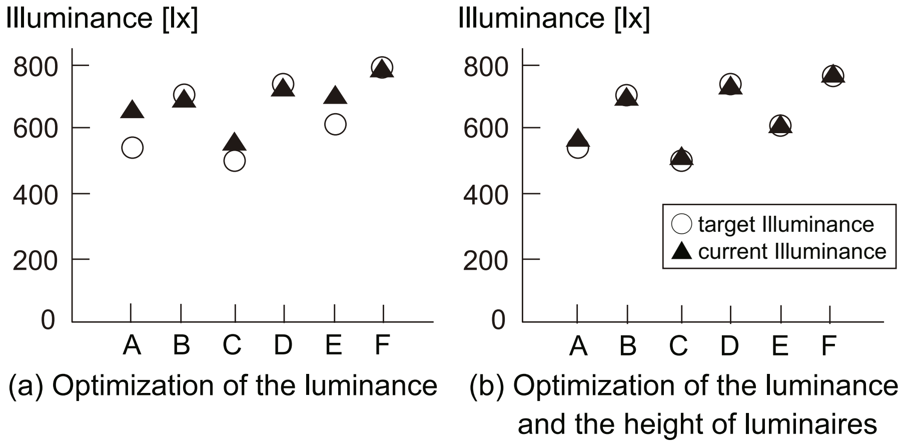
(b) Optimization of the luminance and the height of luminaires

Fig. 3. The case that some workers require extremely high levels of illuminance

In Fig.3-(a), the luminaire height is not changed and high illuminance of 1200 lx is required in the upper left place. All lights illuminate at high luminous intensity in order to achieve the illuminance, but still the requirement of 1200 lx cannot be achieved. In regards to other places, the difference from the required illuminance increases.

On the other hand in Fig.3-(b), the 1200 lx requirement is achieved by optimizing the luminaire height, and the target illuminance is also achieved for all workers. The height of the luminaire closest to the worker requiring high illuminance is significantly reduced from the initial value of 250 cm to 150 cm. In this way, the maximum illuminance can be increased by lowering the luminaire height. Based on the above, making the luminaire height the design variable is effective in the case that some workers require high illuminance levels.

When adjacent workers require significantly different levels of illuminance. A verification experiment in the case that adjacent workers require significantly different levels of illuminance is conducted under the condition of 15 white fluorescent lamps and six workers requiring different illuminance levels. Three low values (500, 540, 600 lx) and three high values (700, 760, 800 lx) of target illuminance are alternatively arranged. As a result of optimizing the luminous intensity for each light without changing the luminaire height under this situation, only three out of six workers' required illuminance is achieved and power consumption for the lights is 62 % of the time when all lights are illuminated at one hundred percent level, as indicated in Fig.4-(a). On the contrary, in the case of optimizing the luminous intensity for each light as well as the luminaire height, illuminance required by all workers is achieved and the power consumption for the light is 52 %, as indicated in Fig.4-(b).



(a) Optimization of the luminance

(b) Optimization of the luminance and the height of luminaires

Fig. 4. Target illuminance and realization illuminance of workers

When the results of Fig.3-(a) and (b) are compared, it is found that illuminance required by workers can be achieved at high accuracy by optimizing the luminaire height. In this way, the target illuminance for all workers is also achieved in the case that adjacent workers require significantly different levels of illuminance, and more energy savings is achieved than in the case that the height is not made the design variable. In accordance with the above, making the luminaire height the design variable is effective to realize significantly different levels of illuminance that could not have been achieved before.

4 Conclusion

In this study, a method is proposed to realize the required illuminance by optimizing not only the luminous intensity but also the height by using height-adjustable luminaires, in the case that distribution of desirable illuminance levels cannot be physically achieved, including the case that adjacent workers require significantly different levels of illuminance from each other. It is found that this method has the effect to increase the maximum illuminance which could not have been achieved before, as well as the effect to satisfy different levels of illuminance which could also not have been achieved before.

References

1. Miki, M., Hiroyasu, T., Imazato, K.: Proposal for an Intelligent Lighting System and Verification of Control Method Effectiveness. In: Proc. IEEE CIS, pp. 520–525 (2004)
2. Press Release from Koyuyo Co., Ltd. (in Japanese), http://www.kokuyo.co.jp/press/news/20081118-889.html
3. Press Release from Mitsubishi Real Estate Co., Ltd. (in Japanese), http://www.mec.co.jp/j/news/pdf/mec090331.pdf
4. Miki, M., Hiroyasu, T., Imazato, K., Yonezawa, M.: Intelligent Lighting Control using Correlation Coefficient between Luminance and Illuminance. In: Proc. IASTED Intelligent Systems and Control, vol. 497(078), pp. 31–36 (2005)
5. Shikakura, T., Morikawa, H., Nakamura, Y.: Perception of Lighting Fluctuation in Office Lighting Environment. J. Light & Vis. Env. 27(2), 75–82 (2003)

RSIE: A Tool Dedicated to Reflexive Systems

Yann Barloy, Jean-Marc Nigro, Sophie Loriette, and Baptiste Cable

Charles Delaunay Institute - UTT, 12 rue Marie Curie, 10000 Troyes, France
{yann.barloy,jean-marc.nigro,sophie.loriette,baptiste.cable}@utt.fr

Abstract. This article deals with how metaknowledge can improve rule-based system and presents a new Reflexive System Inference Engine (RSIE) which enables not only the activation of rules, but also metarules, making it belong to systems managing metaknowledge. The experimentation section shows a rule-based system named IDRES with a structure which has been modified to use metaknowledge.

Keywords: Inference Engine, Reflexive Systems, Metaknowledge.

1 Introduction

The domain of metaknowledge was conceived in the 1970s and 80s [1] at the same time as the emergence of rule-based systems. A metaknowledge can be defined as being knowledge about knowledge. Different classes of metaknowledge were established by Jacques Pitrat [2]: metaknowledge for acquiring, for explaining, for using or for stocking knowledge.

There is no specific architecture (or programming language) for handling metaknowledge. However, rule-based systems have the advantage of enabling the building of different levels of knowledge [3] and the developer can focus on the transcription of methods in the form of rules without being concerned about their triggering.

It is in this context that the idea of a new inference engine, called RSIE [4] (Reflexive System Inference Engine) appeared. This idea had already been introduced in a theoretical way by Clancey [5] or Torsun [6] who had developed a logic allowing the programming of a "meta" level by using a language such as PROLOG. The development of a tool allowing the manipulation of metaknowledge [7][8] would facilitate the implementation of AI systems.

RSIE allows thus the developer to build systems based on rules and meta-rules (a meta-rule is executed in the same way as a rule). Contrary to most systems which use meta-rules [9], [2] in a static way, the triggering of meta-rules can dynamically modify the structure of the rules during the session. Reflective systems can also be made by RSIE.

In the following section, the structure of RSIE is presented. Then, the article deals with the domain of application: CASSICE and more particularly IDRES. Part five presents the structure of the facts and the rules used in RSIE. Finally, the last part shows the results we obtained with and without the use of metaknowledge.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 363–370, 2010.
© Springer-Verlag Berlin Heidelberg 2010

2 The Inference Engine

Two types of inference engines are known: those based on a filter algorithm and those using an RETE network [10]. In order to benefit from the advantages of theses two methods, we use a hybrid method. A new kind of network will be created for each rule from the rule base. The idea is to preserve the powerful aspect of RETE architecture while preventing that a meta-action entails a total rebuilding of the rule base but affects only concerned rules.

This new kind of network, called RSIE-network, allows not only the filtering of facts (for example: $X < 10$) but also parts of rules (for example: attribute of condition 1 in rule 2 = X). Thanks to theses networks it is possible to put a part of a rule as a condition to another one. The advantage is that the system has information about its own knowledge and can consequently adapt the rules which are tested to the current evolution of a problem.

In order to enable this, two bases of rules are necessary. The “active base” contains the rules which have an associated RSIE-network and could be matched. It represents the part of the global knowledge that is used to solve a given problem. The “passive base” contains the others rules. Rules can be moved from one base to the other during the resolution of a problem according to their relevance.

As an example, the Management metarule uses information on the recognition rules in order to manage them in the most effective way during the recognition of the manoeuvre.

Metarule Management
If Rule1 belongs to the overtaking sequence
  Rule1 has number n1 in this sequence
  Rule1 has been matched
Then Move to the active base the rule with the number n1+1 in the sequence
  Move to the waiting base the rule with the number n1-1 in the sequence

This metarule uses meta-conditions and meta-actions.

A meta-condition is a condition on a rule contrary to a simple condition that relates to facts. It may include a rule in its entirety or so on parts of it.

While a normal action acts on the fact base, a meta-action can add, edit or remove rules from a rule base.

This feature can be very useful because it allows to adapt the knowledge used in the problem to solve. Indeed, in a classical system all the knowledge needed to solve the problem are given from the outset and are tested until the end.

Meta-actions can adjust this knowledge according to the status of the problem. For example, by removing from the rule base knowledge that has surely no longer be useful reduces the conflict set which enables the system to be more efficient. Similarly, the rule known in advance that it will be triggered at the end is not necessary in early resolution and may be added in due course.

The possibility of changing rules allow for it to change the rule base. Indeed, if a rule is never triggered because too restrictive, it may be wise to relax the constraints and RSIE can do it while running.

Thus, as would a student who realizes that his reasoning does not lead to the solution and decides to change a system with meta-rules type RSIE may change its approach if the results are not consistent with expectations.

RSIE is written in Java with the IDE NetBeans6 because this language has some reflexive aspects that we make use of.

3 Domain of Application

The aim of the CASSICE [11] project is the realization of a computerized system capable of listing situations of real driving. This project is based on collaboration with many French research laboratories. It uses an Experimental Vehicle (EV) equipped with a camera and a set of proprioceptive sensors (rev-counter, speed, speed of wheels rotation, lighting and road marking, sensor of wheel angle, accelerometer) supplying different values quoted in table 1. Each lines of the table 1 will be called "data lines" in the article. The meaning of each data is presented in the table 2.

Table 1. Acquired data with EV

Time	X	Y	V	Θ	Acc	Φ	Rg	Rd
0.01	32.0	0.0	15	0.00	0	0.00	-3.50	1.50
0.02	31.8	0.0	15	0.00	0	0.00	-3.50	1.50
...	...	...	...	...	...	...	...	...
1.14	15.2	2.1	15	-9.46	0	3.00	-1.41	3.59

Table 2. Meaning of data

Time	Clock (s)
Acc	EV's relative acceleration with regard to TV ($m/s2$)
Φ	Angle of EV's front wheels (degree)
Rd	EV's position with regard to right side of the road (m)
Rg	EV's position with regard to left side of the road (m)
Θ	Angle of the TV (degree)
V	EV's relative speed with regard to TV (m/s)
X	Relative axial position of TV in reference to EV (m)
Y	Relative lateral position in reference to EV (m)

4 IDRES System

IDRES system participates at the CASSICE project to recognize manoeuvres made by the vehicle from a sequence of known data (cf. Table 1). In this article, only the recognition of the overtaking manoeuvre is dealt with. The following principle is adopted:

A manoeuvre is decomposed in a sequence of situations. It is important to respect the order of realization of these situations. So, the overtaking manoeuvre was decomposed into ten states: *Waiting for overtaking, Overtaking intent, Beginning left lane change, Crossing left discontinuous line, End left lane change, Passing, End of passing, Beginning right lane change, crossing right discontinuous line, End right lane change.*

IDRES is operational and gives good results (cf. Fig. 1). Nevertheless, its execution presents some difficulties: the simultaneous treatment of more than 40 lines of acquisitions takes a very long time for giving the results. The system must cut the sequence in intervals of 30 lines of acquisitions and make the recognition of the manoeuvre on every interval.

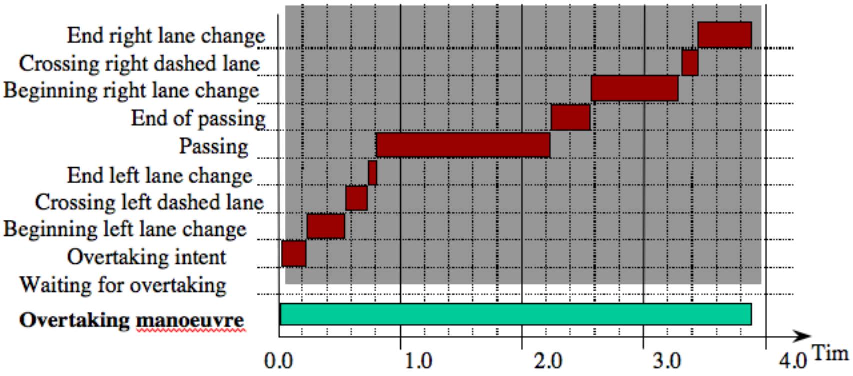


Fig. 1. Overtaking manoeuvre recognition

By looking at the release of IDRES rules, we notice that most of them are useful only during a relatively short lap of time. In addition, they require many resources (memory and time) during the running of inferences engine. To solve this problem, a solution would be to remove these rules when they cannot be executed any more (cf. Fig. 2). The use of RSIE allows the creation of meta-actions which remove some rules when convenient.

The use of these meta-actions in each rule makes the system more effective. The difference between the gray zones of figures 1 and 2 shows a benefit of more than 55% in term of occupation of the resources used for the rules matching.

However, IDRES can still be improved. Indeed, one can see on figure 2 that rules which will only be matched at the end are tested at the beginning of the recognition. To avoid that, it is necessary to give only some initial rules to the system and to create or rather to include in the network progressively relevant rules.

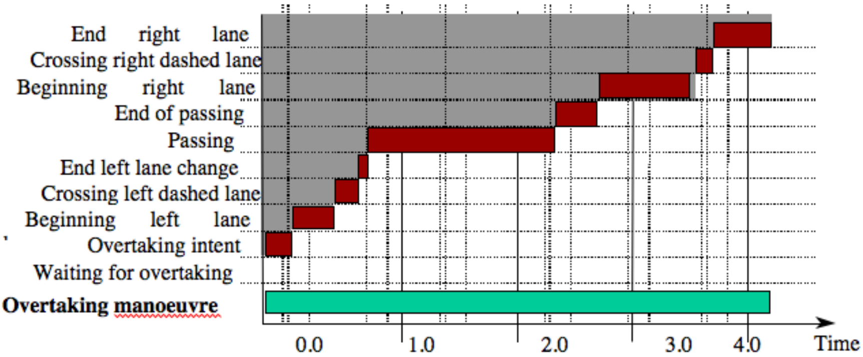


Fig. 2. Validity of rules by the use of meta-actions to delete rules

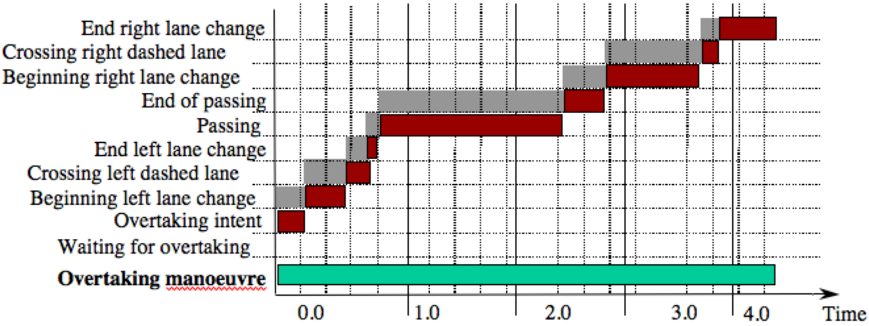


Fig. 3. Validity of rules by the use of meta-actions to add and remove rules

The use of this second kind of meta-action in addition to the first one in each rule makes the system more effective. The differences between the gray zones of figures 1 and 3 increase the benefit in term of occupation of the resources used for the rules matching to more than 80%.

It is also possible to use only one metarule containing meta-actions and meta-conditions. In fact, we can obtain the same results as seen in figure 3 by using "simple rules" in combination with the Metarule Management.

This implies that the system is able to access knowledge that composes a manoeuvre to recognize which rule it is judicious to use at every t moment. This is to allow the design of such systems that RSIE was created.

5 Data Representation

In this section we will present the structure of the facts and the rules which are handled by RSIE.

Structure of a fact:
[Object] [Attribute] [Value]

Structure of a condition:
[Object] [Attribute] [Operator] [Value]

Here [Object] correspond to a time
[Attribute] = X means that we look the car position in X
[Operator] can be=, >, <, >=, <=
[Value] of a fact is compared with the [Value] in a condition

IDRES Rules:

Rule Waiting for overtaking Number = 1 Group = Overtaking
If !time $X < 0$
  !time $Y > -0.9$
Then CreateFact !time State = Waiting_for_overtaking

In the first line, “Waiting for overtaking” is the name of the rule. The other piece of information is that this rule belongs to the “Overtaking” group and that it is the first one in this group.

“!time $X < 0$” searches in the fact base if, at a t moment, the value of the attribute X is lower than 0. In fact, it is when the experimental vehicle is behind the target vehicle.

“!time $Y > -0.9$” searches the facts where the Y value is higher than -0.9. In this case, it means that the two vehicles are on the same lane.

When the system finds two facts with the same [Object] which verify the conditions, it creates a new fact “!time State = Waiting_for_overtaking”. It means that a state of the car has been recognized.

Metarule Management Number = 1 Group = Meta
If #Rule GROUP = Overtaking
  #Rule RULENUMBER = !n1
  #Rule MATCHED = TRUE
Then AddActive !n1+1 Overtaking
    AddWaiting !n1-1 Overtaking

This rule is the same as the one we described at the end of the second section. The difference is that here it is described in RSIE formalism. Its name is “Management” and it is the first one of the group “Meta”.

First of all, it searches for all the rules which belong to the overtaking group in the “active” base of rules. Then, it takes the number of these rules. Finally, it looks if these same rules have been matched.

If a rule verifies all these conditions, the Metarule Management is matched and the action part is executed. It adds to the active base the next rule of the sequence of recognition and moves the rule before from the active base to the waiting one. It activates the “!n1+1” rule and deactivates the “!n1-1”.

6 Experiments

RSIE was tested with a 3.00 GHz Pentium by executing the first IDRES’ level. It includes 10 rules and every 10 ms eight facts are generated from the data of eight sensors. The data were treated line by line. The table above describes the results we obtain for three executions of IDRES with rules that recognize each state of the manoeuvre and with or without the management metarule.

	First execution	Second execution	Third execution
Without metarule	1.078 s	1.016 s	1.129 s
With metarule management	0.632 s	0.651 s	0.640 s

These results show that the use of metaknowledge allows IDRES to match fewer rules. It therefore does fewer tests and manages fewer facts, using therefore less memory space. Indeed, the system creates 821 facts without the meta-level and only 405 with this level.

The use of metarules makes it possible to give only a limited number of initial rules to the system and thus optimizes the speed of execution and the memory capacity.

We can see that the cost of treatment of the metarules remains lower than the benefit generated by their use.

Moreover, if the user of the system wants to recognize another manoeuvre, he just has to change one value in the management metarule. Indeed, if the needed rules are in the waiting rule base and the sequence of the manoeuvre is defined, by changing “overtaking” by “line changing” the system will adapt the knowledge for the new manoeuvre.

7 Conclusion

The structure used to develop and to use RSIE allows the conception of meta-rules, meta-conditions or meta-actions as easily as one rule. It also permits the creation of reflexive meta-rules (which apply to themselves) without having to duplicate knowledge [12]. Another advantage is that the reflexive system inference engine is able to execute these meta-rules during the execution, thus allowing the development of learning techniques in real time or in very dynamic domains (like assistance in driving cars).

RSIE is operational and its main advantage of is its capacity to design systems that make choice based not only on the environment but also on its own knowledge. Such systems know which knowledge is being used and which other can be available and adapted to solve a given problem.

The aim of RSIE's next version is to add learning techniques based on monitoring. In fact, the monitoring is particularly useful to judiciously pick the initial tests, avoid delay in exploring a way that seemed promising and proved disappointing, correct directions initially taken, find any errors, gain expertise: the analysis of fruitful and fruitless trials leads to an apprenticeship, and finally to be autonomous.

A new user interface is also envisaged to make the program more ergonomic.

Aknowlegment. Research supported in part by Champagne-Ardenne Regional Council (district grant) and the European Social Fund.

References

1. Hayes, P.: Computation and deduction. In: Symposium on Mathematical Foundations of Computer Science, pp. 105–118 (1973)
2. Pitrat, J.: An intelligent system must and can observe its own behviour. In: Cognitiva 1990 (1990)
3. Genesereth, M., Nilsson, N.: Logical Fundations of Artificial Intelligence. Morgan Kaufmann, CA (1987)
4. Barloy, Y., Nigro, J.: Rsie: An inference engine for reflexive systems. In: Applications and Innovations in Intelligents Systems XV, pp. 315–320 (2007)
5. Clancey, W.: Model construction operators. Artificial Intelligence 53, 1–115 (1992)
6. Torsun, I.: Foundation of Intelligent Knowledge-Based Systems. Academic Press, London (1995)
7. Laurière, J.: Snark: a langage to represent declarative knowledge and inference engine which use heuristics. In: Information Processing 1986, pp. 811–816. Elsevier Publisher, Amsterdam (1986)
8. Spreeuwenberg, S., Gerrits, R., Boekenoogen, M.: Valens: A knowledge based tool to validate and verify an aion knowledge base. In: ECAI 2000, pp. 731–735 (2000)
9. Cazenave, T.: Metarules to improve tactical go knowledge. Information Sciences 154(3-4), 173–188 (2003)
10. Forgy, C.: Rete: A fast algorithm for the many pattern/many object pattern match problem. Artificial Intelligence 19 (1982)
11. Nigro, J., Barloy, Y.: The meta inference engine: a tool to use metaknowledge. In: IPMU 2006 (2006)
12. Kornman, S.: Infinite regress with self-monitoring. In: Reflection 1996 Conference, San Francisco, pp. 221–233 (1996)

A Model for Temperature Prediction of Melted Steel in the Electric Arc Furnace (EAF)

Marcin Blachnik, Krystian Mączka, and Tadeusz Wieczorek

Silesian University of Technology,
Department of Management and Computer Science ,
Katowice, Krasinskiego 8, Poland

Abstract. A constant aspiration to optimize electric arc steelmaking process causes an increase of the use of advanced analytical methods for the process support. The goal of the paper is to present the way to predict temperature of melted steel in the electric arc furnace and consequently, to reduce the number of temperature measurements during the process. Reducing the number of temperature measurements shortens the time of the whole process and allows increasing production.

1 Introduction and Problem Statement

The electric arc steelmaking process usually consists of three main steps: melting steel scraps in electric arc furnace (EAF), refining the steel in the ladle heating furnace (LHF) and the continuous casting process (CCS) (fig.1). During the EAF phase, the main aim is to melt down metal scrap in the shortest time possible. During the LHF stage, furnace additives are injected to the liquid steel to obtain proper chemical constitution of steel. Continuous casting of steel ends the whole process. In this paper, the authors will consider the possible improvements of the first stage of the process - EAF. Steel production

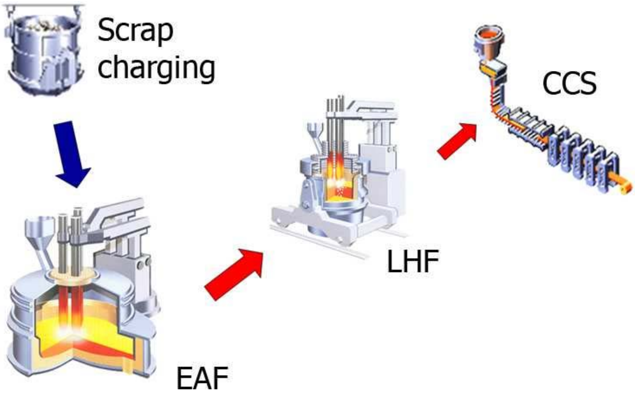


Fig. 1. Diagram of typical electric arc steelmaking process

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 371–378, 2010.
© Springer-Verlag Berlin Heidelberg 2010

by the use of EAF bases only on metal scrap. That is why, EAF has become one of the world's main steel production method. The natural reserves of metal ore are still decreasing and becoming more expensive. Another advantage is the fact that the time of the EAF process is shorter than the time of any other steel production method. (e.g. than blast furnaces). The EAFs capacity and power level have steadily expanded during the last decades. Due to the fact that the main goal of the EAF stage is to melt down metal scrap in the shortest time possible, an oportunity to reduce even one minute of this process is very important.

It has been already mentioned that the metal scrap is the only charge in the EAF process which should be melted in the shortest time possible. Reducing even one minute of this period of time may be very important and may allow saving money. There are several ways to achieve the goal [10,11]: by appropriate management of the process (e.g. impedance matching of the current track and the feedstock of the furnace, chemical composition, etc.) or by reducing the time and the number of periods when the electric arc is switched off (what is not a trivial problem) [12]. The last approach is considered in this paper.

The type of scrap most frequently used is, so-called, merchant scrap, which consists of a variety of elements [4]. The industrial practice and market situation shows that there is still uncertainty about determining the melting process parameters (e.g. steel temperature). The most obvious method to identify steel temperature in the furnace is observation of the EAF process. However, because of high temperature, high dust density, flames (which are composed of a variety of combustion gases, where each of them absorbs light to varying degrees and different wavelengths) and other circumstances in the EAF, a reliable observation of the process is almost impossible. Recently, papers referring to the first tests of direct observations of the melting process in an EAF have appeared [5]. For this purpose, camera-based technology for monitoring the scrap melting process in the EAF has been developed. However, this method presently has not got any practical applications. Another way of experimental identification of the temperature in the EAF is a continuous temperature measuring. Results of the first tests appeared in the paper [6] and presented a development of temperature measuring system, which showed that there is a possibility to accurately measure temperature of steel in the furnace for the whole furnace cycle.

Currently, the temperature is measured a few times during overheating period with the use of the thermocouple. The furnace operator turns off the electric arc and manually places the disposable thermocouple in the liquid steel. This process of measurement takes at least two minutes so reducing the number of temperature measurements may reduce the time of the whole process.

Another way to identifiy temperature in the EAF, is the intelligent modeling [1,2], which is the main aim of this paper. The authors present a research concerning building the temperature prediction model. The method is based on the fact that in the last phase of the EAF process, during the overheating period, two or even three temperature samples are taken. Until this moment, there is no information about liquid steel temperature. By proper prediction of temperature of liquid steel, one can reduce the energy comsumption during the EAF process.

It is important to remember that every temperature measurement takes about two or three minutes, but in the context of a sixty minute process it is a long time and the posibility of reducing even one temperature measurement is very important. Reducing only one temperature measurement in each melting process could increase steel production nearly thousands of tones per year [8,9].

In the paper, the authors will consider aspects of data acquisition and also data preparation, which are very important factors for proper model learning.

2 Dataset

The first and one of the most important steps of building the model is data acquisition, which will be used to train the model. Accuracy of the model strongly depends on the quality of the dataset delivered to data mining tools. Appropriate data preprocessing is crucial to build a model that will have a small error rate. Understanding of real process, which is described by the data is a very important aspect during building the dataset. It allows rejecting data that is inappropriate from the technological point of view and which could be wrongly registered. By knowing technical details of data collection procedures many pieces of information can be obtained, such as possible redundancy, oversampling (which appeared in our calculations), etc. Such knowledge can be also used to preselect decision model or narrow down a problem of model selection.

In this section, the authors describe the process of gathering and preparing data for calculations and also describe the whole research environment.

2.1 Data Source

Data used for training the model came from the real metallurgical process. The data describes scrap melting process in the electric arc furnace (EAF). The data was collected in one of Polish steelworks by the use of Simatic-5 and Simatic-7 programmable logic controllers (PLC) which factory uses for process control and for recording process parameters. Over 200 variables incoming from controlers every second are recorded by *Data acquisition server* (fig.2) which communicates with all PLC's and saves values of variables into the Industrial SQL Server (InSQL).

Dataset was collected in two different periods. The first dataset was gathered after six months' work of InSQL server. That dataset was used as *training set* for building the temperature prediction model. That training set consists of 2127 samples. Then, the second dataset was collected for model validation (*validation set*). It includes description of over 1200 EAF melting processes of the last 8 months and over 3500 temperatures measurements taken during those melting processes. The collected dataset describes processes of a production of over 40 different grades of steel.

For these calculations, the most important variables were those describing weight of the first, the second and the third scrap charging basket; electric energy used for melting each basket; electric energy used during the overheating period; amount of blowed oxygen and solid carburizing and temperatures of thermocouples located in furnace bottom. The last, the 17th variable, was temperature of melted steel, whose prediction was our aim.

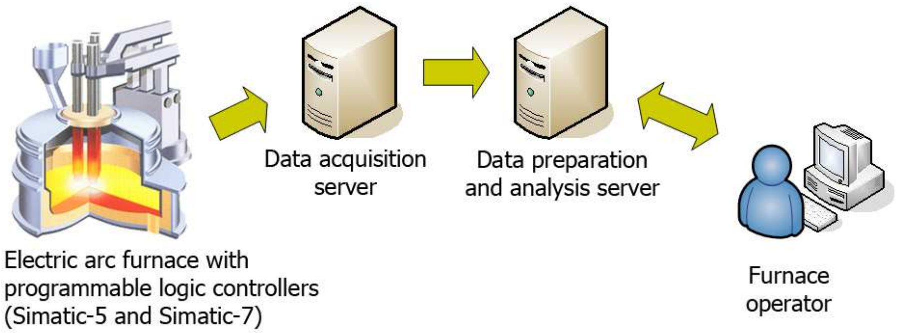

Data acquisition server
Data preparation and analysis server
Electric arc furnace with programmable logic controllers (Simatic-5 and Simatic-7)
Furnace operator

Fig. 2. Diagram of real environment data flow

Many variables not used directly in calculations were also extremely important for the process analysis, identification of particular steps of the process and for proper data acquisition. There were, amongst other things: primary and secondary voltage, primary and secondary current, transformer tap position, power coefficient, positions of electrodes etc.

2.2 Data Preparation

From data collected in the database we have selected following attributes:

- *Time* - exact data and time of the measurement
- *MeltNumber* - ID number of a melt
- *Mass*$_1$, *Mass*$_2$, *Mass*$_3$ - Mass of each steel scrap basket
- *Energy*$_1$ ... *Energy*$_3$ - Electric energy consumed during melting of each backet
- *CurrentEnergy* - Electric energy consumed during overheating period (from the end of melting the third basket until now)
- *TemperatureSensor*$_1$ ... *TemperatureSensor*$_6$ - Values of temperature measured by sensors placed inside the bottom of the furnace
- *Temperature* - manually measured temperature by the furnace staff

Basing on these attributes, after outlier elimination by the use of interquartile range, two different datasets were built. The first one (*dataset A*) was created for direct temperature prediction. In this dataset,the variables that are described above were taken to analysis and *temperature* was marked as the output variable. All input variables were normalized to keep its values in the range $[0, 1]$. The second dataset (*dataset B*) was defined as a difference between the current and the previous measurement of the following attributes:

- $dTime = Time_{i+1} - Time_i$ - time passed between measurements
- $dCurrentEnergy = CurrentEnergy_{i+1} - CurrentEnergy_i$ - Electric energy consumed during $dTime$ period
- $dTemperatureSensor_1 = TemperatureSensor_{1,i+1} - TemperatureSensor_{1,i}$ ... $dTemperatureSensor_6$=$TemperatureSensor_{6,i+1}$–$TemperatureSensor_{6,i}$ - A change of temperature values measured by sensors during $dTime$ period

- $dTemperature = Temperature_{i+1} - Temperature_i$ - Changes in the temperature of the steel bath during the $dTime$ period

The final dataset was extended by the *mass* attribute.

Both datasets have its advantages and disadvantages. *dataset A* was very simple to implement in real environment however, it was sensitive to external parameters and to the behaviour of the whole process of steel scrap melting. We believe that this dataset might not be accurate enough to fulfill the desired requirements because melting process is very unstable by nature and external, unpredictable behaviour, caused by other stages of steel production (e.g. the breaks in CCS stage force breaks in EAF) might appear. In contrast to *dataset A* , *dataset B* wouldn't be useful until the first temperature is measured. In other words, it requires at least one single temperature measurement because the prediction model was able to predict only a change of the temperature, so the final temperature was calculated as the sum of the last measured temperature and the temperature change.

$$temp = temp_{last} + \Delta_{temp} \tag{1}$$

where:

- $temp$ - predicted temperature
- $temp_{last}$ - last measured temperature
- Δ_{temp} - predicted temperature change

We believe that this dataset is more stable (less sensitive to unpredictability of the process) and leads to better results because most of the attributes were calculated as the differences of their values.

3 Building the Model

The goal of the prediction model is to achieve the best possible accuracy [3], what in our case leads to the smallest temperature prediction error, measured by the use of mean square error (MSE). To find the best possible regression model, various algorithms were evaluated on the datasets described in the previous section. In our calculations, we considered following regression algorithms:

- simple linear regression (LR)
- SVM for regression with ϵ-insensitive cost function and linear (SVR$_L$) and Gaussian kernel (SVR$_G$)
- kNN regression algorithm

Both SVM algorithm and kNN require the tuning of parameters. All the parameters were selected by 10-fold cross validation test and by the use of the most common greed-search strategy. For kNN, k-value was considered as $k = [1 \ldots 10]$ while SVM required searching in quadratic for linear kernel and cubic space for Gausian kernel [7]. For SVM, the following values were tested: softness parameter $C = 2^{[-3-11,3,5,7]}$, cost function parameter $\epsilon = [15, 10, 7, 5, 3, 1, 0.1, 0.01]$ and for Gaussian kernel γ = [0.5, 0.7, 1, 1.3, 1.5].

Table 1. Comparison of the MSE error rate of LR, kNN, SVR$_L$ and SVR$_G$ models

	dataset A		*dataset B*	
Model	MSE	ME	MSE	ME
LR	1092.97 ± 37.96	762.10 ± 48.04	31.77 ± 2.05	25.97 ± 2.32
kNN	32.14 ± 3.48	23.97 ± 1.76	19.51 ± 1.37	15.29 ± 1.13
SVR$_L$	29.69 ± 2.28	21.65 ± 1.13	18.13 ± 1.72	14.02 ± 1.48
SVR$_G$	40.39 ± 4.68	28.93 ± 1.71	18.06 ± 1.43	14.06 ± 1.3

The obtained results for the best set of parameters are reported in the table (1).

The analysis of results presented in the table (1) proved our assumptions related to both datasets. System requirements indicated that only results obtained for *dataset B* were acceptable while the results obtained for *dataset A* were charged with too large error, not acceptable in real environment. Moreover, the value of the temperature is necessary only during overheating period to determine the proper moment for the begining of casting.

Surprisingly, linear regression in comparison with SVR$_L$ obtained much worse results, though, both models were linear. It is worth a mention that there is a small difference in accuracy between the linear and non-linear (Gaussian kernel based) SVM model. According to this fact, SVR$_L$ was implemented in the production system.

4 Validation in Real Environment

The best predicting model was used in the real environment. For this purpose, proper implementation for *Data preparation and analysis server (DPAS)* was prepared (fig.2). This server was running Microsoft SQL Server, where proper server jobs were responsible for: transmition of suitable data from *Data acquisition server*; identification of the particular steps of the process; collecting and preparing data for calculation model; recording answers of predicting model and sending temperature prediction to the furnace operator.

Operator, during the last step of the process (during overheating), was supported by model prediction and could make a decision whether to take or not to take another temperature measurement. Each real measurement was recorded by *DPAS* in its database and that allowed for the comparison of the results.

In the figure 3, the obtained prediction error is presented. A large majority of prediction values were in the error range from -20 to +20 $^\circ C$. However, there were over 67 cases from 3500 predictions compared with real measurements, where error was greater than 50 Celsius degrees.

Table 2. Statistics of obtain error values

	Maximum	Minimum	Mean	Standart deviation
Temperature error value $^\circ C$	154*	1.1	4.4	21

*The maximum error occurred one time among 3500 predictions and probably was caused by wrong measurement.

The maximim, minimum, mean and standard deviation of the occurred error (in $^\circ C$) is presented in the table 2.

5 Conclusions and Further Research

Prediction of the liquid steel temperature during the EAF process is a very important task. The value of steel temperature determines the proper moment for the begining of casting. The results obtained in our experiments proved that it is possible to achieve much better accuracy of prediction by adequate dataset preparation. The results obtained with *dataset A* have an error rate by one-third higher from those obtained with *dataset B* . We believe that the reason is that *dataset B* was built basing on the differences of values of variables, which allows reducing significantly the impact of unstable behaviour of the initial melting phases. The best of all evaluated models was SVM model that allowed obtaining temperature error accuracy, which was 18 Celsius degrees. In our experiments both SVM models (SVR$_L$ and SVR$_G$) got similar accuracy. However, the SVR$_G$ gave slightly better results with smaller value of variance. However, in real environment the linear model (SVR$_L$) was implemented because it was much simpler and faster than the nonlinear one.

After eight months of using that model, we were able to compare the obtained results with values of real temperature measurements. The provided results proved a very good quality of our model. Several outliers appeared (fig.3), however they could be caused by inconsistence of the validation set or by the wrong real measurement. Taking consideration of the obtained results and the results presented in the table (2), we came to

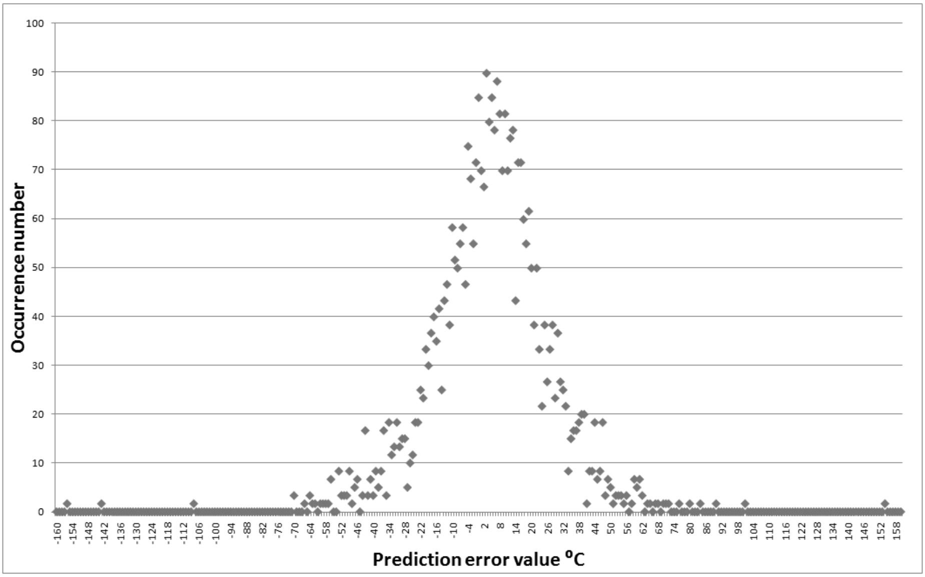


Fig. 3. Histogram of prediction error

the conclusion that mean error of our model does not equal zero. We believe that this fact is related to the *concept shift*, caused by the changes in the production procedures that appeared during the last eight months.

We believe that this model can be further improved. There are several ways to achieve this goal. Firstly, we did not take into consideration energy delivered to the furnace by the flames of the burners and the gases. Taking energy into account, may significantly improve accuracy of the model. We also plan to extend the dataset by even more variables and we plan to apply feature selection methods to determine automatically the most important variables. The last concept is to improve the data preprocessing instead of calculating the difference between the following samples and map them to another, more sophisticated space.

References

1. Wieczorek, T., Blachnik, M., Mączka, K.: Modelowanie procesu roztapiania złomu w piecu łukowym z wykorzystaniem sieci neuronowych i algorytmów SVM. In: Grosman, F., Hyrcza-Michalska, M. (eds.) Informatyka w technologii metali, pp. 161–168. Wydawnictwo Naukowe Akapit, Kraków (2008)
2. Wieczorek, T., Mązka, K.: Modeling of the AC-EAF process using computational intelligence methods. Electrotechnical Review 11, 184–188 (2008)
3. Wieczorek, T., Blachnik, M., Mączka, K.: Building a model for time reduction of steel scrap meltdown in the electric arc furnace (EAF): General strategy with a comparison of feature selection methods. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 1149–1159. Springer, Heidelberg (2008)
4. Wieczorek, T., Pilarczyk, M.: Classification of steel scrap in the EAF process using image analysis methods. Archives of Metallurgy and Materials 53(2), 613–618 (2008)
5. Millman, M.S., Nyssen, P., Mathy, C., Tolazzi, D., Londero, L., Candusso, C., Baumert, J.C., Brimmeyer, M., Gualtieri, D., Rigoni, D.: Direct observation of the melting process in an eaf with a closed slag door. Archives of Metallurgy and Materials 53(2), 463–468 (2008)
6. Kendall, M., Thys, M., Horrex, A., Verhoeven, J.P.: A window into the electric arc furnace, a continuous temperature sensor measuring the complete furnace cycle. Archives of Metallurgy and Materials 53(2), 451–454 (2008)
7. Schölkopf, B., Smola, A.: Learning with Kernels. In: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge (2001)
8. Gerling, R., Louis, T., Schmeiduch, G., Sesselmann, R., Sieber, A.: Optimizing the melting process at AC-EAF with neural networks. Metall. Jg. 53(7-8), 410–418 (1999)
9. Pappe, T., Obradovic, D., Schlang, M.: Neural networks: reducing energy and raw materials requirements. Simens Review, 24–27 (Fall 1995)
10. Baumert, J.-C., Engel, R., Weiler, C.: Dynamic modeling of the electric arc furnace process using artificial neural networks. In: La Revue de Metallurgie-CIT, vol. 10, pp. 839–849 (2002)
11. Boulet, B., Lalli, G., Ajersch, M.: Modeling and control of an electric arc furnace. In: Proc. of the American Control Conf., Denver, Colorado, pp. 3060–3064. IEEE Press, Los Alamitos (2003)
12. Gao, X., Li, S., Chai, T., Shao, C., Wang, X.: Set point intelligent optimal control of electric arc furnace. In: Proc. of 2nd Asian Control Conf., Seul, pp. 763–766 (1997)

Parallel Hybrid Metaheuristics for the Scheduling with Fuzzy Processing Times*

Wojciech Bożejko[1], Michał Czapiński[2], and Mieczysław Wodecki[2]

[1] Institute of Engineering, Wrocław University of Technology
Janiszewskiego 11-17, 50-372 Wrocław, Poland
wojciech.bozejko@pwr.wroc.pl

[2] Institute of Computer Science, University of Wrocław
Przesmyckiego 20, 51-151 Wrocław, Poland
mwd@ii.uni.wroc.pl

Abstract. In this paper, parallel simulated annealing with genetic enhancement algorithm (HSG) is presented and applied to permutation flow shop scheduling problem which has been proven to be $\mathcal{NP}$-complete in the strong sense. The metaheuristics is based on a clustering algorithm for simulated annealing but introduces a new mechanism for dynamic SA parameters adjustment based on genetic algorithms. The proposed parallel algorithm is based on the master-slave model with cooperation. Fuzzy arithmetic on fuzzy numbers is used to determine the minimum completion times $C_{\max}$. Finally, the computation results and discussion of the algorithms performance are presented.

1 Introduction

Practical machine scheduling problems are numerous and varied. They arise in diverse areas such as flexible manufacturing systems, production planning, communication, computer design, etc. A scheduling problem is to find sequences of jobs on given machines with the objective of minimizing some function. In a simpler version of the problem, flow shop scheduling, all jobs pass through all machines in the some order. In this paper, we deal with another special version of the problem called a permutation flow shop (PFS) scheduling problem where each machine processes the jobs in the same order. The PFS problem belongs to the NP-hard class problems, however a solution of such a problem is usually made using heuristic approach that converges to a locally optimal solution.

In recent studies, scheduling problems were fuzzificated by using the concept of fuzzy due date and processing times. In paper Dumitru and Luban [3] investigate the application of fuzzy sets on the problem of the production scheduling. Tsujimura et al. [12] present the branch and bound algorithm for the three machine flow shop problem when job processing times are described by triangular fuzzy numbers. Especially fuzzy logic application on the scheduling problems (by using fuzzy processing times) is presented in papers: Ishibuschi and Murata [6], Izzettin and Serpil [5] and Peng and Liu [9].

[*] The work was supported by MNiSW Poland, within the grant No. N N514 232237.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 379–386, 2010.
© Springer-Verlag Berlin Heidelberg 2010

In this study, flow shop scheduling problem of the typical situation of the flexible production systems which occupy a very important place in recent production systems are taken into consideration with fuzzy processing time.

2 Flow Shop Scheduling

The permutation flow shop problem can be formulated as follows. Each of n jobs from the set $J = \{1, 2, \ldots, n\}$ has to be processed on m machines $1, 2, \ldots, m$ in that order. Job $j \in J$, consists of a sequence of m operations; operation O_{jk} corresponds to the job j processing on machine k during an uninterrupted processing time p_{jk}. Assumptions:

(a) for each job only one operation can be processed on a machine,
(b) each machine can process only one job at a time,
(c) the processing order is the same on each machine
(d) all jobs are available for machine processing simultaneously at time zero.

We want to find a schedule such that the processing order is the same on each machine and the maximum completion time is minimal.

The flow shop problem is NP-complete and thus it is usually solved by approximation or heuristic methods. The use of simulated annealing is presented, e.g., in Osman and Potts [8], Bożejko and Wodecki [1] (parallel algorithm), tabu search in Nowicki and Smutnicki[7], Grabowski and Wodecki [4], and genetic algorithm in Reeves [11].

Each schedule of jobs can be represented by the permutation $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$ on the set J. Let Π denote the set of all such permutations. We wish to find such permutation $\pi^* \in \Pi$, that

$$C_{\max}(\pi^*) = \min_{\pi \in \Pi} C_{\max}(\pi),$$

where $C_{\max}(\pi)$ is the time required to complete all jobs on the machines.

3 Flow Shop Scheduling with Fuzzy Processing Times

Let us suppose that processing times of the jobs on machines are not deterministic but they are given by fuzzy numbers.

In this paper the fuzzy processing times $p_{i,j}$ $(i = 1, 2, \ldots, m,\ j = 1, 2, \ldots, n)$ are represented by a triangular membership function μ (i.e. 3-tuple $\widetilde{p}_{i,j} = (p_{i,j}^{\min}$, $p_{i,j}^{\text{med}}$, $p_{i,j}^{\max})$ $(i = 1, 2, \ldots, m,\ j = 1, 2, \ldots, n)$ with the following properties:

i) $(p_{i,j}^{\min} \le p_{i,j}^{\text{med}} \le p_{i,j}^{\max})$,
ii) $\mu(a) = 0$ for $a \le p_{i,j}^{\min}$ or $a \ge p_{i,j}^{\max}$,
iii) $\mu(p_{i,j}^{\text{med}}) = 1$,
iv) μ is increasing on $[p_{i,j}^{\min}, p_{i,j}^{\text{med}}]$ and decreasing on $[p_{i,j}^{\text{med}}, p_{i,j}^{\max}]$.

The addition of fuzzy numbers $\widetilde{a} = (a_1, a_2, a_3)$ and $\widetilde{b} = (b_1, b_2, b_3)$, can be derived from the extension principle and it is as follows (see [2])

$$\widetilde{a} + \widetilde{b} = (a_1 + b_1, a_2 + b_2, a_3 + b_3).$$

Similarly

$$\max\{\widetilde{a}, \widetilde{b}\} = (\max\{a_1, b_1\}, \max\{a_2, b_2\}, \max\{a_3, b_3\}).$$

If the time of the job execution $\pi(i)$ ($\pi \in \Pi$) on the machines j is determined by a fuzzy number

$$\tilde{p}_{\pi(i),j} = (p^{\min}_{\pi(i),j},\ p^{\text{med}}_{\pi(i),j},\ p^{\max}_{\pi(i),j}),$$

then its finishing time is a fuzzy number in the form of:

$\tilde{C}_{\pi(i),j} = \left(C^{\min}_{\pi(i),j}, C^{\text{med}}_{\pi(i),j}, C^{\max}_{\pi(i),j}\right)$, where $C^{\min}_{\pi(i),j}$, $C^{\text{med}}_{\pi(i),j}$ and $C^{\max}_{\pi(i),j}$ can be determined from the following recurrent formulas:

$$C^{\delta}_{\pi(i),j} = \max\{C^{\delta}_{\pi(i-1),j},\ C^{\delta}_{\pi(i),j-1}\} + p^{\delta}_{\pi(i),j}, \quad \delta \in \{\min, \text{med}, \max\},$$

with the initial conditions

$$C^{\delta}_{\pi(0),j} = 0 \;\; j = 1, 2, \,\ldots, m, \;\; C^{\delta}_{\pi(i),0} = 0 \;\; i = 1, 2, \,\ldots, n, \quad \delta \in \{\min, \text{med}, \max\}$$

The time of all jobs execution (in the π sequence) is also a fuzzy number

$$\tilde{C}_{\max}(\pi) = (C^{\min}_{\pi(n),m},\ C^{\text{med}}_{\pi(n),m},\ C^{\max}_{\pi(n),m}).$$

The ranking function defined as follows is to compare fuzzy cost function values:

$$\Im(\tilde{C}_{\max}(\pi)) = \frac{1}{4}\left(C^{\min}_{\pi(n),m} + C^{\text{med}}_{\pi(n),m} + C^{\text{med}}_{\pi(n),m} + C^{\max}_{\pi(n),m}\right) \tag{1}$$

The permutation flow shop scheduling problem with fuzzy processing times (FPFS) consists in determining a permutation $\pi^* \in \Pi$ such that

$$\Im(\tilde{C}_{\max}(\pi)) = \min\{\Im(\tilde{C}_{\max}(\beta)) :\ \beta \in \Pi\},$$

which fulfills constrains (a)–(d).

4 Parallel Hybrid Algorithm

In this section simulated annealing (SA) with the genetic enhancement algorithm (HSG)is used for the permutation flow shop problem with $C_{\max}$. The classic SA algorithm and all modifications leading to HSG are described below. We shall present methods of algorithms parallelization as well as its modifications for the flow shop problem in which execution times are fuzzy numbers.

Classic simulated annealing algorithm

In classic simulated annealing in each iteration a new solution is generated and evaluated. If it is better than the original solution it is accepted, and if it is worse then it is accepted with probability equals to $\exp(-\delta/T)$, where δ is the difference between values of the original and the new solution, and T is a control parameter corresponding to temperature in annealing process in metallurgy. The SA algorithm general scheme is presented on the listing:

```
T := start temperature current := generate initial solution
evaluate current
best:=current
repeat
   count:=0
   repeat
      candidate:=generate candidate from current
      evaluate candidate
      if candidate is better than best then update best
      accept candidate as current with probability
         equal to min(1, exp((F(current) - F(candidate))/T)
      count++
   until (count == number of iterations at temperature T)
   decrease temperature according to cooling scheme
until (stopping criteria)
```

To generate new candidate solutions both transposition and insertion moves are applied with equal probability. In HSG basic geometric cooling scheme is used, so the temperature decreases according to the formula $T_{new} = \alpha \times T_{old}$.

Clustering algorithm for simulated annealing
In their paper, Ram et al. [10] propose a parallel Clustering algorithm for simulated annealing (CASA). In this algorithm master and worker nodes are distinguished. CASA is divided into generations in which the master node distributes the initial solution (or solutions) along worker nodes so they can start running simulated annealing independently. Then, each fixed number of iterations of SA, best solutions found by worker nodes are gathered by the master node. Next the generation starts with the best solution found so far as the initial solution. This model of parallelism is adopted into the HSG algorithm.

Simulated annealing with genetic enhancement
As it has been mentioned before, a mechanism to dynamically adjusted SA's configuration during the runtime is introduced in order to reduce its influence on performance. The SA configuration includes start temperature, minimal temperature and cooling ratio. Number of SA iterations between consecutive temperature reductions is computed in such a way that at the end of the generation temperature is equal to the minimal temperature.

Algorithm description, configuration and complexity
The master node is responsible for generating initial solution. Then at beginning of each *generation*, the master node broadcasts the best solution found so far if necessary (i.e. in first generation or when new best solution was found in a previous generation). After that, worker nodes start SA algorithm for a fixed number of iterations and master node waits until they finish. Finally, the master node gathers values of the best solutions found by worker nodes and selects the best among them. If this best value is better than the best value stored by the

master node, respective permutation is received from a worker node which had found it, and this solution is considered as the best in the next generation. Below listing contains pseudo-code for the the master node in the HSG algorithm.

```
best := generate initial solution
for i := 1 to number of generations do
   broadcast best to worker nodes (if necessary)
   gather values of the best solutions found by worker nodes
   select best solution and receive permutation
   from a worker node that found it (if necessary)
   update best
end.
```

At each worker node a fixed number of generations is performed. Each of these generations starts with receiving the best solution from the master node (if necessary). This solution is then used as an initial solution to execute SA algorithm for a fixed number of iterations. Each worker node starts with different SA configuration (referred to as *individual*, in analogy to genetic algorithms) which is generated randomly with uniform distribution from following ranges: $[1, 200]$ for start temperature, $[0.1, 1]$ for minimal temperature and $[0.9, 1)$ for cooling ratio. After SA algorithm is finished, value of the best solution is returned to the master node. If this value is the best among other worker nodes, and better than the best value stored by the master node, respective permutation is sent back to the master node.

Each individual has its TTL (time to live) with initial value which is HSG's parameter. If a solution returned by SA algorithm at the end of generation is not better than initial one, TTL of respective worker's individual is reduced by one. Otherwise, TTL is reset to the initial value. If individual survives (i.e. its TTL is positive), limit for minimal temperature is lifted and worker continues next generation with a temperature, which it has finished last generation with. If TTL of any individual reaches zero, it is replaced with a new one, generated randomly. Listing contains pseudo-code for each worker node in the HSG algorithm and Figure 1 presents co-operation between nodes in the HSG algorithm.

```
Generate inital individual (SA configuration)
   for i:=1 to number of generations do
      receive best solution from the master node (if necessary)
      execute SA algorithm for a fixed number of iterations
      send back the best value found to the master node
      send back the best solution's permutation
         to the master node (if necessary)
      update individual's TTL and replace it if TTL reaches zero
   end.
```

The HSG's configuration includes number of worker nodes, number of generations and number of SA iterations to be performed in each generation and initial TTL for each individual.

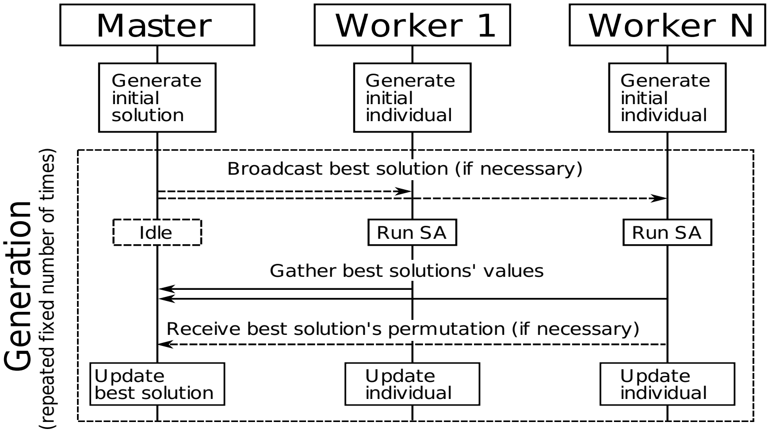

Master
Worker 1
Worker N
Generate initial solution
Generate initial individual
Generate initial individual
Generation
(repeated fixed number of times)
Broadcast best solution (if necessary)
Idle
Run SA
Run SA
Gather best solutions' values
Receive best solution's permutation (if necessary)
Update best solution
Update individual
Update individual

Fig. 1. Co-operation between nodes in HSG algorithm

5 Hybrid Algorithm with Fuzzy Processing Times

In the Section 3 the method of cost function value calculation for the permutation flow shop problem with fuzzy jobs execution times was described. The adequately modified HSG algorithm (in which fuzzy jobs execution times was included) will be represented as FzHSG.

5.1 Algorithms Stability

Let $\mathbf{p} = [p_{i,j}]_{nxm}$ be (deterministic) jobs execution times for an instance of the PFS problem. By $\mathcal{D}(\mathbf{p})$ we describe a set of examlpes of data generated from $\mathbf{p}$ by the disturbance of jobs execution times (i.e. elements from $\mathbf{p}$). The disturbance consists in random changes of $p_{i,j}$ values, $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$. This procedure is detailed described in the Section 5.2. We use the following notion:

$\mathcal{A}$	- an algorithm of solving PFS problem,
$\mathbf{p} = [p_{i,j}]_{nxm}$	- an instance of data (execution times) for the PFS problem,
$\pi_{\mathbf{p}}^{\mathcal{A}}$	- a solution determined by the algorithm $\mathcal{A}$ for the data $\mathbf{p}$,
$C_{\max}(\pi_{\mathbf{p}}^{\mathcal{A}}, \mathbf{d})$	- a value of the cost function for the data $\mathbf{d}$ and a sequence of jobs execution (permutation) $\pi_{\mathbf{p}}^{\mathcal{A}}$.

Let $\mathbf{p}$ be an instance of deterministic data and $\mathcal{D}(\mathbf{p})$ a set of disturbed data. For an algorithm $\mathcal{A}$ and an instance of disturbed data $\mathbf{d}$

$$\delta(\mathcal{A}, \mathbf{p}, \mathbf{d}) = \frac{C_{\max}(\pi_{\mathbf{p}}^{\mathcal{A}}, \mathbf{d}) - C_{\max}(\pi_{\mathbf{d}}^{\mathcal{A}}, \mathbf{d})}{C_{\max}(\pi_{\mathbf{d}}^{\mathcal{A}}, \mathbf{d})} \cdot 100\%. \tag{2}$$

This formula defines a percentage relative deviation of the cost function value for the $\mathbf{d}$ if jobs are executed in the sequence $\pi_{\mathbf{p}}^{\mathcal{A}}$ and $\pi_{\mathbf{d}}^{\mathcal{A}}$. By

$$\Delta(\mathcal{A}, \mathbf{p}, \mathcal{D}(\mathbf{p})) = \frac{1}{|\mathcal{D}(\mathbf{p})|} \sum_{\mathbf{d} \in \mathcal{D}(\mathbf{p})} \delta(\mathcal{A}, \mathbf{p}, \mathbf{d}) \tag{3}$$

we define the ***stability of the best solution*** of an instance $\mathbf{p}$ determined by an algorithm $\mathcal{A}$ on the set of disturbed data $\mathcal{D}(\mathbf{p})$. Permutations $\pi_{\mathbf{p}}^{\mathcal{A}}$ and $\pi_{\mathbf{d}}^{\mathcal{A}}$ are the best solutions determined by the algorithm $\mathcal{A}$ for the data $\mathbf{p}$ and $\mathbf{d} \in \mathcal{D}(\ \mathbf{p})$, respectively.

Let Ω be a set of some (deterministic) data instances for the PFS problem. The ***algorithm stability*** $\mathcal{A}$ on the data set Ω

$$\mathbf{S}(\mathcal{A}, \Omega) = \frac{1}{|\Omega|} \sum_{\mathbf{p} \in \Omega} \Delta(\mathcal{A}, \mathbf{p}, \mathcal{D}(\mathbf{p}))). \tag{4}$$

5.2 Computational Experiments

The algorithms HSG and FzHSG were coded in C++ using MPICH2 implementation of MPI standard for communication between nodes. All experiments were ran on Cranfield Univerity's Astral cluster, which is equipped with 856 Xeon 3 GHz processors, 2 GB memory for each, and Infiniband network. HSG and FzHSG tested on the first 6 groups of benchmark instances (see OR Library: http://mscmga.ms.ic.uk/info.html). The benchmark set contains 120 particulary hard instances of 12 size. For each size (group, ta001-ta060) $n \times m$: 20×5, 20×10, 20×20, 50×5, 50×10, 50×20.

Fuzzy jobs execution times generation

If $p_{i,j}$ $(i = 1, 2, \ldots, m,\ j = 1, 2, \ldots, n)$ is an instance of deterministic data for the PFS problem, then fuzzy jobs execution times $\widetilde{p}_{i,j}$ are represented by a triple $(p_{i,j}^{\min}, p_{i,j}^{\text{med}}, p_{i,j}^{\max})$, where

$$p_{i,j}^{\min} = \max\{1, \lceil p_{i,j} - p_{i,j}/3 \rceil\},\ \ p_{i,j}^{\text{med}} = p_{i,j},\ \text{ and }\ p_{i,j}^{\max} = \lceil p_{i,j} + p_{i,j}/6 \rceil.$$

Disturbed data generation

For each instance of deterministic data $\mathbf{p} = \{p_{i,j}\}_{m \times n}$ there were 100 instances generated - elements of the set $\mathcal{D}(\mathbf{p})$. If an instance of the data $\mathbf{d} \in \mathcal{D}(\mathbf{p})$ $(\mathbf{d} = \{d_{i,j}\}_{m \times n})$ than jobs execution times were drawn (due to the uniform distribution) from the range $[\max\{1, \lceil p_{i,j} - p_{i,j}/3 \rceil\}, \lceil p_{i,j} + p_{i,j}/6 \rceil]$.

There were 6 000 instances generated of disturbed data in total. For the each group of instances the values parameters (2) and (3) were calculated and they are shown in the Table 1. Values $\delta_{\min}$ i $\delta_{\max}$ are values of the minimal and maximal deviation defined by (2), respectively, for the group of instances, and δ_{aprd} – an average value. The stability of both algorithm was also calculated on the set Ω including groups of instances shown in the Table 1. For the HSG algorithm, $\mathbf{S}(\text{HSG}, \Omega) = 2.02\%$, and for the algorithm with fuzzy jobs execution times FzHSG, $\mathbf{S}(\text{FzHSG}, \Omega) = 2.00\%$. Therefore, the stability of both algorithms is almost identical. It follows among others from this that fuzzy jobs finishing times (after defuzzification) were insignificantly different from the times calculated for the input deterministic times. The disturbance process of the jobs execution times makes the times of jobs shorter or longer, but it has small influence on the time of jobs finishing. Therefore, values of the cost function are almost identical. The application of parallelism makes possible to execute all the calculations in about 1 hour.

Group	*Parallel hybrid algorithm* HSG			*Fuzzy parallel hybrid algorithm* FzHSG		
	δ_{min}	δ_{aprd}	δ_{max}	δ_{min}	δ_{aprd}	δ_{max}
20×5	0.022	1.782	5.487	0.022	1.892	5.471
20×10	0.157	2.466	6.013	0.102	2.326	5.964
20×20	0.270	2.246	5.263	0.389	2.275	5.254
50×5	0.044	1.238	4.088	0.030	1.240	4.150
50×10	0.222	2.172	4.577	0.255	2.196	5.122
50×20	0.528	2.198	4.155	0.541	2.178	4.219

6 Conclusions

We present a parallel simulated annealing algorithm with genetic enhancement for a permutation flow shop problem with fuzzy processing times. The algorithm introduces a dynamic parameters adjustment for simulated annealing algorithm. The parallel algorithm stability was also defined and researched for the deterministic and fuzzy jobs execution times.

References

1. Bożejko, W., Wodecki, M.: Solving the Flow Shop Problem by Parallel Simulated Annealing. In: Wyrzykowski, R., Dongarra, J., Paprzycki, M., Waśniewski, J. (eds.) PPAM 2001. LNCS, vol. 2328, pp. 236–244. Springer, Heidelberg (2002)
2. Dubois, D., Prade, H.: Theorie des Possibilites. In: Applications a la representation des connaissances en informatique. MASSON, Paris (1988)
3. Dumitru, V., Luban, F.: Membership functions, some mathematical programming models and production scheduling. Fuzzy Sets and Systems 8, 19–33 (1982)
4. Grabowski, J., Wodecki, M.: A very fast tabu search algorithm for the permutation flow shop problem with makespan criterion. Computers & Operations Research 31, 1891–1909 (2004)
5. Izzettin, T., Serpil, E.: Fuzzy branch-and-bound algorithm for flow shop scheduling. Journal of Intelligent Manufacturing 15, 449–454 (2004)
6. Ishibuschi, H., Murata, T.: Scheduling with Fuzzy Duedate and Fuzzy Processing Time. In: Słowiński, R., Hapke, M. (eds.) Scheduling Under Fuzziness, pp. 113–143. Springer, Heidelberg (2000)
7. Nowicki, E., Smutnicki, C.: A fast tabu search algorithm for the permutation flowshop problem. European Journal of Operational Research 91, 160–175 (1996)
8. Osman, I., Potts, C.: Simulated Annealing for Permutation Flow-Shop Scheduling. OMEGA 17(6), 551–557 (1989)
9. Peng, J., Liu, B.: Parallel machine scheduling models with fuzzy processing times. Information Sciences 166, 49–66 (2004)
10. Ram, J.D., Sreenivas, T.H., Subramaniam, G.K.: Parallel simulated annealing algorithms. Journal of Parallel and Distributed Computing 37(2), 207–212 (1996)
11. Reeves, C.: A Genetic Algorithm for Flowshop Sequencing. Computers & Operations Research 22(1), 5–13 (1995)
12. Tsujimura, Y., Park, S.H., Change, I.S., Gen, M.: An effective method for solving flow shop problems with fuzzy processing times. Computers and Industrial Engieering 25(1-4), 239–242 (1993)

A Neuro-tabu Search Algorithm for the Job Shop Problem*

Wojciech Bożejko and Mariusz Uchroński

Institute of Computer Engineering, Control and Robotics
Wrocław University of Technology, Janiszewskiego 11-17, 50-372 Wrocław, Poland
{wojciech.bozejko,mariusz.uchronski}@pwr.wroc.pl

Abstract. This paper deals with tabu search with neural network instead of classic tabu list applied for solving the classic job shop scheduling problem with makespan criterion. Computational experiments are given and compared with the results yielded by the best algorithms discussed in the literature. These results show that the proposed algorithm solves the job shop instances with high accuracy in a very short time. Presented ideas can be applied for many scheduling problems.

1 Introduction

The paper deals with the job shop problem which can be briefly presented as follows. There is a set of jobs and a set of machines. Each job consists of a number of operations which have to be processed in a given order, each one on a specified machine during a fixed time. The processing of an operation cannot be interrupted. Each machine can process at most one operation at a time. We want to find a schedule (the assignment of operations to time intervals on machines) that minimizes the *makespan*.

The job shop scheduling problem, although relatively easily stated, is NP-hard and it is considered as one of the hardest problems in the area of combinatorial optimization. Many various methods have been proposed, ranging from simple and fast dispatching rules to sophisticated branch-and bound and metaheuristic algorithms. For the literature see Balas and Vazacopoulos [1] (guided local search method with shifting bottleneck), Morton and Pentico [5] (heuristic local search), Nowicki and Smutnicki [6] (tabu search with representatives and block properties), Vaessens et al. [9] (local search methods), and their references.

In a classic tabu search method a move is chosen in each iteration of the algorithm. This move is remembered on the list of the length *maxt* called the *tabu list* and it is forbidden for *maxt* number of iteration. After executing *maxt* iterations by the algorithm this move is removed from the list and it can be executed again. One can say that this move looses its status of being forbidden 'suddenly'. Here we present a mechanism in which a status a forbidden move is changing in the exponential way. Such an approach was successfully applied for the quadratic assignment problem [4] and for the flow shop scheduling problem [7].

* The work was supported by MNiSW Poland, within the grant No. N N514 232237.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 387–394, 2010.
© Springer-Verlag Berlin Heidelberg 2010

2 Problem Formulation and Preliminaries

The job shop problem can be formally defined as follows, using the notation by Nowicki and Smutnicki [6]. There are: a set of jobs $J = \{1, 2, ..., n\}$, a set of machines $M = \{1, 2, ..., m\}$, and a set of operations $O = \{1, 2, ..., o\}$. Set O decomposes into subsets (chains) corresponding to the jobs. Each job j consists of a sequence of o_j operations indexed consecutively by $(l_{j-1}+1, ..., l_{j-1}+o_j)$, which are to be processed in order, where $l_j = \sum_{i=1}^{j} o_i$, is the total number of operations of the first j jobs, $j = 1, 2, ..., n$, $(l_0 = 0)$, and $o = \sum_{i=1}^{n} o_i$. Operation x is to be processed on machine $\mu_x \in M$ during processing time p_x, $x \in O$. The set of operations O can be decomposed into subsets $M_k = \{x \in O | \mu_x = k\}$, each containing the operations to be processed on machine k, and $m_k = |M_k|$, $k \in M$. Let permutation π_k define the processing order of operations from the set M_k on machine k, and let Π_k be the set of all permutations on M_k. The processing order of all operations on machines is determined by m-tuple $\pi = (\pi_1, \pi_2, ..., \pi_m)$, where $\pi \in \Pi_1 \times \Pi_2 \times ... \times \Pi_m$.

It is useful to present the job shop problem by using a graph. For the given processing order π, we create the graph $G(\pi) = (N, R \cup E(\pi))$ with a set of nodes N and a set of arcs $R \cup E(\pi)$, $N = O \cup \{s, c\}$, where s and c are two fictitious operations representing dummy 'start' and 'completion' operations, respectively. The weight of node $x \in N$ is given by the processing time p_x, $(p_s = p_c = 0)$. The set R contains arcs connecting consecutive operations of the same job, as well as arcs from node s to the first operation of each job and from the last operation of each job to node c. Arcs in $E(\pi)$ connect operations to be processed by the same machine. Arcs from set R represent the processing order of operations in jobs, whereas arcs from set $E(\pi)$ represent the processing order of operations on machines. The processing order π is feasible if and only if graph $G(\pi)$ does not contain a cycle.

Let $C(x, y)$ and $L(x, y)$ denote the longest (critical) path and length of this path, respectively, from node x to y in $G(\pi)$. It is well-known that makespan $C_{max}(\pi)$ for π is equal to length $L(s, c)$ of critical path $C(s, c)$ in $G(\pi)$. Now, we can rephrase the job shop problem as that of finding a feasible processing order $\pi \in \Pi$ that minimizes $C_{max}(\pi)$ in the resulting graph.

We use a notation similar to the paper of Balas and Vazacopoulos [1]. For any operation $x \in O$, we will denote by $\alpha(x)$ and $\gamma(x)$ the job-predecessor and job-successor (if it exists), respectively, of x, i.e. $(\alpha(x), x)$ and $(x, \gamma(x))$ are arcs from R. Further, for the given processing order π, and for any operation $x \in O$, we will denote by $\beta(x)$ and $\delta(x)$ the machine-predecessor and machine-successor (if it exists), respectively, of x, i.e. the operation that precedes x, and succeeds x, respectively, on the machine processing operation x. In other words, $(\beta(x), x)$ and $(x, \delta(x))$ are arcs from $E(\pi)$.

Denote the critical path in $G(\pi)$ by $C(s, c) = (s, u_1, u_2, \ldots, u_w, c)$, where $u_i \in O$, $1 \le i \le w$, and w is the number of nodes (except fictitious s and c) in this path. The critical path $C(s, c)$ depends on π, but for simplicity in notation we will not express it explicitly. The critical path is decomposed into subsequences $B_1, B_2, \ldots, B_r$ called *blocks* in π on $C(s, c)$ (Grabowski et al. [3]), where

1. $B_k = (u_{f_k}, u_{f_k+1}, ..., u_{l_k-1}, u_{l_k}), \quad 1 \leq f_k \leq l_k \leq w, \quad k = 1, 2, ..., r.$
2. B_k contains operations processed on the same machine, $k = 1, 2, \ldots, r$.
3. two consecutive blocks contain operations processed on different machines.

In other words, the block is a maximal subsequence of $C(s, c)$ and contains successive operations from the critical path processed consecutively on the same machine. In the further considerations, we will be interested only in *non-empty* block, i.e. such that $|B_k| > 1$, or alternatively $f_k < l_k$. Operations u_{f_k} and u_{l_k} in B_k are called the *first* and *last* ones, respectively. The k-th block, exclusive of the first and last operations, is called the k-th *internal block.*

A block has advantageous so-called *elimination properties*, introduced originally in the form of the following theorem (Grabowski et al. [3]).

Theorem 1. *Let $G(\pi)$ be an acyclic graph with blocks B_k, $k = 1, 2, ..., r$. If acyclic graph $G(\omega)$ has been obtained from $G(\pi)$ through the modifications of π so that $C_{max}(\omega) < C_{max}(\pi)$, then in $G(\omega)$*

(i) at least one operation $x \in B_k$ precedes job u_{f_k}, for some $k \in \{1, 2, ..., r\}$, or
(ii) at least one operation $x \in B_k$ succeeds job u_{l_k}, for some $k \in \{1, 2, ..., r\}$.

Example 1. There are three jobs, 9 operations and three machines, $n = 3$, $m = 2$, $o = 9$. The job 1 consist of the sequence of three operations (1,2,3), the job 2 consist of a sequence of three operations (4,5,6) and the job 3 consist of a sequence of three operations (7,8,9). Operations have to be processed on machines $\mu_2 = \mu_6 = \mu_7 = 1$, $\mu_1 = \mu_5 = \mu_9 = 2$, $\mu_3 = \mu_4 = \mu_8 = 3$. A feasible processing order is $\pi = (\pi_1, \pi_2, \pi_3)$, where $\pi_1 = (7, 2, 6)$, $\pi_2 = (1, 5, 9)$ and $\pi_3 = (4, 8, 3)$. The graph $G(\pi)$ is shown in the Figure 2 and the Gantt chart – in the Figure 1.

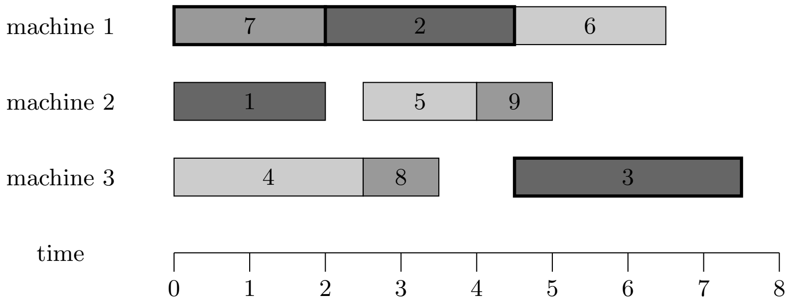


Fig. 1. The Gantt chart for the Example 1

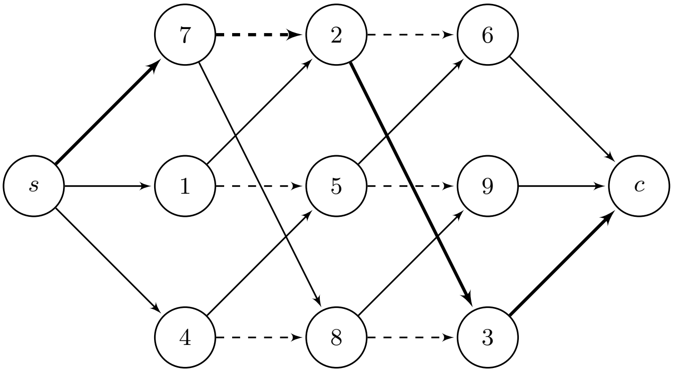


Fig. 2. The graph $G(\pi)$ for the Example 1

3 Tabu Search Method

The tabu search method was proposed by Glover [2]. Generally, it consists of improving the starting solution's value π^*. An algorithm generates the neighborhood of the current solution and seeks the solution which has the minimal value of $C_{\max}(\beta)$, $\beta \in \mathcal{N}(\pi^*)$. This solution β is the starting solution in the next iteration of the algorithm. Such a procedure allows the possibility of increasing the current solution's value (when a new starting solution is sought), but it increases the chance of finding the global minimum. To prevent generating of recently considered solutions (making cycles), those solutions are recorded on a list of prohibited solutions, the so-called tabu list T (short-term memory). A standard tabu search algorithm can be written as follows.

Algorithm 1. Standard tabu search algorithm

```
Let $\pi \in \Pi$ be an initial solution;
        $\pi^*$ – the best known solution;  $\pi^* \leftarrow \pi$ – starting solution;
Step 1. Generate the neighborhood $\mathcal{N}(\pi)$ of the current solution $\pi$.
          Exclude from $\mathcal{N}(\pi)$ elements from the $T$ list except $\beta \in \mathcal{N}(\pi)$ such that
                $C_{\max}(\beta) < C_{\max}(\pi^*)$;
Step 2. Find the solution $\delta \in \mathcal{N}(\pi)$ such, that
                $C_{\max}(\delta) = min(C_{\max}(\beta), \beta \in \mathcal{N}(\pi))$;
Step 3. if $C_{\max}(\delta) < C_{\max}(\pi^*)$ then
                 $\pi^* \leftarrow \delta$;
          Include $\delta$ in the list $T$;  $\pi \leftarrow \delta$;
Step 4. if (Stop condition is true) then Stop; else go to Step 1;
```

Let B_k $(k = 1, 2, \ldots, m)$ be the k-th block in a solution π, B_k^f and B_k^l the subblocks. For job $j \in B_k^f$ by $N_k^f(j)$ let us denote a set of solutions created by moving job j to the beginning of block B_k (before the first job in block $\pi(f_k)$). Analogously, for job $j \in B_k^f$ by $N_k^l(j)$ let us denote a set of solutions created by moving job j to the end of block B_k (after the last job in block $\pi(l_k)$). The neighborhood of the solution π: $\mathcal{N}(\pi) = \bigcup\limits_{j \in B_k} (N_k^f(j) \cup N_k^l(j))$.

Additionally, there is a backtracking mechanism applied in the algorithm (long-term memory). A certain number of good solutions are recorded on backtracking list. Good solution – this means that the relative difference between this solution β and the best known (current) solution $\pi^*$is small or negative – less then ϵ parameter ($\frac{C_{\max}(\beta) - C_{\max}(\pi^*)}{C_{\max}(\pi^*)} < \epsilon$). If there is no improvement of the best solution's objective function value after some number of iterations, the algorithm jumps to the latest solution obtained from the backtracking list (so the current solution π is overwritten by the solution from the list). The current tabu list is also overwritten – the algorithm receives the tabu list connected with the backtracked solution from the backtracking list.

4 Tabu Search Mechanism with Neural Network Application

In the considered tabu search algorithm each move is represented by its neuron. For the neighborhood considered in [6] a network of neurons formed of $o - 1$ neurons. Let i-th neuron represents a move consisting in swap of two adjacent elements on the positions i and $i+1$ in a solution π. In a proposed neural network architecture a history of each neuron is stored as its internal state (*tabu effect*). If in an iteration neuron is activated, then the value 1 is fixed on its output and values 0 are fixed on the outputs of other neurons. The neuron activated in an iteration must not be activated once again for the next s iterations. Each neuron is defined by the following equations:

$$\eta_i(t+1) = \alpha \mathit{\Delta}_i(t),\ \mathit{\Delta}_i(t) = \frac{C_{max}(\pi_v^{(t)}) - C^*_{max}}{C^*_{max}},\ \gamma_i(t+1) = \sum_{d=0}^{s-1} k^d x_i(t-d), \quad (1)$$

where $x_i(t)$ is an output of the neuron i in the iteration t. Symbol $C_{max}(\pi_v^{(t)})$ means the value of the goal function for the permutation obtained after executing a move v in the iteration t, i.e. $\pi^{(t)}$. Symbol $\mathit{\Delta}_i(t)$ means a normalized, current value of the goal function, and C^*_{max} is the value of the best solution found so far. Parameters α i k are scale factors. A symbol $\eta_i(t+1)$ (*gain effect*) defines quality of a move v. A variable $\gamma_i(t+1)$ (*tabu effect*) stores a history of the neuron i for the last s iterations. Neuron is activated if it has a low value of the tabu effect and it gives a better reduction of the C_{max}. More detailed a neuron i is activated if it has the lowest $\{\eta_i(t+1) + \gamma_i(t+1)\}$ value of all the neurons.

If $0 < k < 1$ i $s = t$ then the formula of γ_i from the equation (1) takes the form of:

$$\gamma_i(t+1) = k\gamma_i(t) + x_i(t), \quad (2)$$

where $\gamma_i(0) = 0$ and $x_i(0) = 0$ for each i. From the equation (2) it follows, that the value of $\gamma_i(t)$ of each neuron decreases exponentially (see Figure 3).

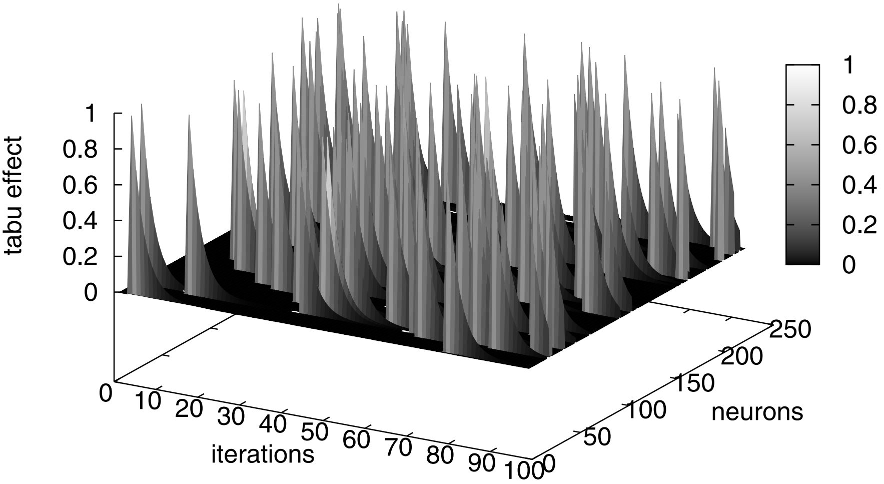


Fig. 3. Changes of the $\gamma(t)$ value

In many algorithms proposed in the literature which are based on the tabu search method a so-called aspiration criterion is implemented. It consists in executing of the forbidden move if it follows to the base solution with the goal function value lower than the best found so far.

In the proposed neuro-tabu search such a function can be implemented by ignoring tabu effect for a move v for which $\Delta_i < 0$. However during computational experiments we have observed that it does not give good effects. Therefore a proposed neuro-tabu search has not got such a function.

5 Computational Experiments

A neuro-tabu search (NTS) algorithm for the job shop scheduling problem was implemented in C++ language and executed on PC with processor with $1.7GHz$ clock. Computational experiments have been provided for the Taillard [8] benchmark instances. Starting solutions of the NTS were determined by using INSA (*INSertion Algorithm*)[6]. The NTS algorithm was terminated after performing 100000 iterations, the value of tuning parameter *max_iter* was taken from [6], *max_iter* = 10000 if makespan has been improved and *max_iter* = 6000 if the backtracking jump has been performed.

The time per single iteration was approximately $3.04 \cdot 10^{-7} \cdot o$ seconds on PC with Intel Celeron 1.7 GHz processor. TSAB was run on PC 386DX which is 430 times slower than PC with Intel Celeron 1.7 GHz processor. The computing power of PC with Intel Celeron 1, 7 GHz was measured using SiSoftware Sandra (the System ANalyser, Diagnostic and Reporting Assistant)[10]. Value of computing power of PC 386DX was taken from [11]. Therefore, in order to normalize

Table 1. Percentage relative deviations to the best known solutions

			NTS		
problem	$n \times m$	INSA	$k = 0.5$	$k = 0.6$	$k = 0.7$
TA01-10	15 × 15	14.62	1.30	1.41	1.30
TA11-20	20 × 15	18.29	2.50	2.65	2.44
TA21-30	20 × 20	17.17	2.36	2.17	2.43
TA31-40	30 × 15	21.13	2.65	1.92	2.29
TA41-50	30 × 20	23.01	3.72	3.70	3.73
TA51-60	50 × 15	16.43	0.09	0.09	0.09
TA61-70	50 × 20	20.07	0.29	0.36	0.22
TA71-80	100 × 20	15.21	0.01	0.01	0.01
average		18.24	1.62	1.54	1.56

Table 2. A percentage relative deviations to the reference solutions given from Taillard [8]

				NTS		
problem	$n \times m$	INSA	TSAB	$k = 0.5$	$k = 0.6$	$k = 0.7$
TA01-10	15 × 15	13.93	0.8	0.70	0.81	0.72
TA11-20	20 × 15	16.50	0.9	0.95	1.10	0.90
TA21-30	20 × 20	15.75	1.2	1.11	0.93	1.18
TA31-40	30 × 15	18.78	0.6	0.68	-0.03	0.33
TA41-50	30 × 20	20.35	1.9	1.48	1.45	1.49
TA51-60	50 × 15	16.38	0.0	0.05	0.05	0.05
TA61-70	50 × 20	17.12	-2.0	-2.16	-2.09	-2.22
TA71-80	100 × 20	15.13	-0.1	-0.05	-0.05	-0.05
average		16.74	0.41	0.35	0.27	0.30

time per single iteration, the algorithm execution time for TSAB was divided by the transformation factor 430. After the normalization the time per single iteration for TSAB equals $0.62 \cdot 10^{-7} \cdot o$ seconds.

The best known solutions, as well as solutions from [8], were taken as reference solutions. The NTS algorithm was executed for various values of the scaling parameter k ($k = 0.5, 0.6, 0.7$) and compared with the best up to now algorithm TSAB of Nowicki and Smutnicki [6]. The value of the k parameter has got a great influence onto obtained results. The best results was obtained when the value of scaling parameter $k = 0.6$ (see Table 1 and Table 2). The table 2 shows the differences between results of TSAB and NTS algorithms – results obtained by NTS algorithms are better than results obtained by TSAB. For instances of the size 30 × 15 the difference between results obtained by NTS and TSAB is the most significant (see Table 2). Percentage relative deviation to the reference solutions for NTS equals -0.03% and for TSAB 0.6%. Change scaling parameter k for instances of the size 50 × 15 and 100 × 20 (Table 1 and Table 2) does not give any effect.

6 Conclusions

We present a fast algorithm based on the neuro-tabu search approach. Computational experiments are provided and compared with the results yielded by the best algorithms discussed in the literature. Results obtained by the proposed algorithm are comparable with the results of the state-of-the-art algorithm TSAB algorithm after a small number of iterations.

As a future work it is possible to adapt the proposed approach for the job shop problem with different types of neighborhood. a neural network with $(o-1)^2$ neurons can be applied for neighborhood which is generated by insertion moves. In this case a neuron (i,j) represents the move of inserting operation $\pi(i)$ in position j. In the traditional tabu search algorithm a move has the tabu status or it has not. The proposed neural network can be also modified by introduce moves with different degrees of tabu.

References

1. Balas, E., Vazacopoulosi, A.: Guided local search with shifting bottleneck for job shop scheduling. Management Science 44(2), 262–275 (1998)
2. Glover, F., Laguna, M.: Tabu Search. Kluwer Academic Publishers, Boston (1997)
3. Grabowski, J., Nowicki, E., Smutnicki, C.: Block algorithm for scheduling of operations in job shop system. Przeglad Statystyczny 35, 67–80 (1988) (in Polish)
4. Hasegawa, M., Ikeguchi, T., Aihara, K.: Exponential and chaotic neuro-dynamical tabu searches for quadratic assignment problems. Control and Cybernetics 29, 773–788 (2000)
5. Morton, T., Pentico, D.: Heuristic scheduling systems. Wiley, New York (1993)
6. Nowicki, E., Smutnicki, C.: A fast taboo search algorithm for the job shop problem. Management Science 42, 797–813 (1996)
7. Solimanpur, M., Vrat, P., Shankar, R.: A neuro-tabu search heuristic for the flow shop scheduling problem. Computers and Operations Research 31, 2151–2164 (2004)
8. Taillard, E.: Benchmarks for basic scheduling problems. European Journal of Operational Research 64, 278–285 (1993)
9. Vaessens, R., Aarts, E., Lenstra, J.K.: Job shop scheduling by local search. INFORMS Journal of Computing 8, 303–317 (1996)
10. http://www.sisoftware.net
11. http://www.roylongbottom.org.uk/mips.htm#anchorAltos

Parallel Meta^{2}heuristics for the Flexible Job Shop Problem*

Wojciech Bożejko[1], Mariusz Uchroński[1], and Mieczysław Wodecki[2]

[1] Institute of Computer Engineering, Control and Robotics
Wrocław University of Technology
Janiszewskiego 11-17, 50-372 Wrocław, Poland
{wojciech.bozejko,mariusz.uchronski}@pwr.wroc.pl
[2] Institute of Computer Science, University of Wrocław
Joliot-Curie 15, 50-383 Wrocław, Poland
mwd@ii.uni.wroc.pl

Abstract. In this paper we consider a double-level metaheuristic optimization algorithm. The algorithm proposed here includes two major modules: the machine selection module which is executed sequentially, and the operation scheduling module executed in parallel. On each level a metaheuristic algorithm is used, so we call this method meta^{2}heuristics. We carry out computational experiment using Graphics Processing Units (GPU). It was possible to obtain new the best known solutions for the benchmark instances from the literature.

1 Introduction

The flexible job shop problem which is considered here constitutes a generalization of the classic job shop problem where operations have to be executed on one machine from a set of dedicated machines. Then, as a job shop problem it also belongs to the strongly NP-hard class. Exact algorithms based on a disjunctive graph representation of the solution have been developed (see Pinedo [9]) but they are not effective for instances with more than 20 jobs and 10 machines. However, many approximate algorithms, mainly metaheuristic, have been proposed. Dauzère-Pérès and Pauli [3] used the tabu search approach extending the disjunctive graph representation for the classic job shop problem to take into consideration assigning operations to machines. Also Mastrolilli and Gambardella [6] proposed a tabu search procedure with effective neighborhood functions for the flexible job shop problem. Many authors have proposed a method of assigning operations to machines and then determining sequence of operations on each machines. Such an approach is followed by Brandimarte [2]. Also genetic approaches have been adopted to solve the flexible job shop problem (Pezzella et al. [8]). Gao et al. [4] proposed hybrid genetic and variable neighborhood descent algorithm for this problem. In this paper we propose a parallel double-level tabu search metaheuristic for the flexible job shop problem. We apply INSA [7] and TSAB [7] algorithm on the second level of parallelism.

* The work was supported by MNiSW Poland, within the grant No. N N514 232237.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 395–402, 2010.
© Springer-Verlag Berlin Heidelberg 2010

2 Flexible Job Shop Problem

The flexible job shop problem (FJSP), also called the general job shop problem with parallel machines, can be formulated as follows. Let $\mathcal{J} = \{1, 2, \dots, n\}$ be a set of jobs which have to be executed on machines from the set $\mathcal{M} = \{1, 2, \dots, m\}$. There exists a partition of the set of machines into types, i.e. subsets of machines with the same functional properties. A job constitutes a sequence of some operations. Each operation has to be executed on an adequate type of machine in a fixed time. The problem consists in the jobs allocation to machines from the adequate type and the schedule of jobs execution determination on each machine to minimize the total jobs finishing time. The following constrains have to be fulfilled:

- (*i*) each job has to be executed on only one machine of a determined type in each moment of time,
- (*ii*) machines must not execute more than one job in each moment of time,
- (*iii*) there are no idle times (i.e. the job execution must not be broken),
- (*iv*) the technological order has to be obeyed.

Let $\mathcal{O} = \{1, 2, \dots, o\}$ be the set of all operations. This set can be partitioned into sequence which correspond to jobs where the job $j \in \mathcal{J}$ is a sequence of o_j operations which have to be executed in an order on dedicated machines (i.e. in so-called technological order). Operations are indexed by numbers $(l_{j-1} + 1, \dots, l_{j-1} + o_j)$ where $l_j = \sum_{i=1}^{j} o_i$ is the number of operations of the first j jobs, $j = 1, 2, \dots, n$, where $l_0 = 0$ and $o = \sum_{i=1}^{n} o_i$.

The set of machines $\mathcal{M} = \{1, 2, \dots, m\}$ can be partitioned into q subsets of the same type where i-th $(i = 1, 2, \dots, q)$ type $\mathcal{M}^i$ includes m_i machines which are indexed by numbers $(t_{i-1} + 1, \dots, t_{i-1} + m_i)$, where $t_i = \sum_{j=1}^{i} m_j$ is the number of machines in the first i types, $i = 1, 2, \dots, q$, where $t_0 = 0$ and $m = \sum_{j=1}^{m} m_j$.

An operation $v \in \mathcal{O}$ has to be executed on the machines type $\mu(v)$, i.e. on one of the machines from the set $\mathcal{M}^{\mu(v)}$ in the time $p_{v,j}$ where $j \in \mathcal{M}^{\mu(v)}$.

Let $\mathcal{O}^k = \{v \in \mathcal{O} : \ \mu(v) = k\}$ be a set of operations executed in the k-th $(k = 1, 2, \dots, q)$ set of machines types. A sequence of operations sets $\mathcal{Q}^k = [\mathcal{Q}^k_{t_{k-1}+1}, \mathcal{Q}^k_{t_{k-1}+2}, \dots, \mathcal{Q}^k_{t_{k-1}+m_k}]$, such that $\mathcal{O}^k = \bigcup_{i=t_{k-1}+1}^{t_{k-1}+m_k} \mathcal{Q}^k_i$ and $\mathcal{Q}^k_i \cap \mathcal{Q}^k_j = \emptyset$, $i \neq j$, $i, j = t_{k-1} + 1, t_{k-1} + 2, \dots, t_{k-1} \ +m_k$ we call an *assignment* of operations to machines in the i-th set of machines types. In a special case a machine can execute no operations and then a set of operations assigned to execute by this machine is an empty set.

A sequence $\mathcal{Q} = [\mathcal{Q}^1, \mathcal{Q}^2, \dots, \mathcal{Q}^q]$, where $\mathcal{Q}^i$ $(i = 1, 2, \dots, q$ is an assignment in the i-th set of machines types we call an *assignment of operations of the set $\mathcal{O}$ to machines from the set $\mathcal{M}$.*

If the assignment of operations to machines has been carried out, then the optimal schedule of operations execution determination (including a sequence of operations execution on machines) boils down to solving the classic scheduling problem, it means the job shop problem.

Let $K = (K_1, K_2, ..., K_m)$ be a sequence of sets where $K_i \in 2^{\mathcal{O}^i}$, $i = 1, 2, ..., m$ (in particular case elements of this sequence can constitute empty sets). By $\mathcal{K}$ we denote the set of all such sequences. The number of elements of the set $\mathcal{K}$ is $2^{|\mathcal{O}^1|} \cdot 2^{|\mathcal{O}^2|} \cdot \ldots \cdot 2^{|\mathcal{O}^q|}$.

If $\mathcal{Q}$ is an assignment of operations to machines then $\mathcal{Q} \in \mathcal{K}$ (of course, the set $\mathcal{K}$ includes also sequences which are not feasible; that is such sequences do not constitute assignments of operations to machines).

For any sequence of sets $K = (K_1, K_2, ..., K_m)$ $(K \in \mathcal{K})$ by $\Pi_i(K)$ we denote the set of all permutations of elements from K_i. Thereafter, let $\pi(K) = (\pi_1(K), \pi_2(K), ..., \pi_m(K))$ be a concatenation m sequences (permutations), where $\pi_i(K) \in \Pi_i(K)$. Therefore $\pi(K) \in \Pi(K) = \Pi_1(K) \times \Pi_2(K) \times, ..., \Pi_m(K)$. It is easy to observe that if $K = (K_1, K_2, ..., K_m)$ is an assignment of operations to machines then the set $\pi_i(K) (i = 1, 2, ..., m)$ includes all permutations (possible sequences of execution) of operations from the set K_i on the machine$i$. Further, let $\Phi = \{(K, \pi(K)) : K \in \mathcal{K} \wedge \pi(K) \in \Pi(K)\}$. Any feasible solution of the FJSP is a pair $(Q, \pi(Q)) \in \Phi$ where $\mathcal{Q}$ is an assignment of operations to machines and $\pi(Q)$ is a permutations concatenation determining the operations execution sequence which are assigned the each machine fulfilling constrains (i-iv).

By Φ° we denote a set of feasible solutions for the FJSP. Of course $\Phi^\circ \subset \Phi$.

3 Graph Representation

Any feasible solution $\Theta = (\mathcal{Q}, \pi(\mathcal{Q})) \in \Phi^\circ$ (where $\mathcal{Q}$ is an assignment of operations to machines and $\pi(\mathcal{Q})$ determines the operations execution sequence on each machine) of the FJSP can be shown as a directed graph with weighted vertexes $G(\Theta) = (\mathcal{V}, \mathcal{R} \cup \mathcal{E}(\Theta))$ where $\mathcal{V}$ is a set of vertexes and a $\mathcal{R} \cup \mathcal{E}(\Theta)$ is a set of arcs, whereas:

1) $\mathcal{V} = \mathcal{O} \cup \{s, c\}$, where s and c are additional (fictitious) operations which represents 'start' and 'finish', respectively. A vertex $v \in V \setminus \{s, c\}$ has two attributes:
   - $\lambda(v)$ – a number of machine on which an operation $v \in \mathcal{O}$ has to be executed,
   - $p_{v,\lambda(v)}$ – a weight of the vertex which equals to the time of operation $v \in \mathcal{O}$ execution on the assigned machine $\lambda(v)$.

   Weights of additional vertexes ($p_s = p_c = 0$).

2) $\mathcal{R} = \bigcup_{j=1}^{n} \left[\bigcup_{i=1}^{o_j - 1} \{(l_{j-1} + i, l_{j-1} + i + 1)\} \cup \{(s, l_{j-1} + 1)\} \cup \{(l_{j-1} + o_j, c)\} \right].$

   A set $\mathcal{R}$ includes arcs which connect successive operations of the job, arcs from the vertex s to the first operation of each job and arcs from the last operation of each job to the vertex c.

3) $\mathcal{E}(\Theta) = \bigcup_{k=1}^{m} \bigcup_{i=1}^{|\mathcal{O}^k|-1} \{(\pi_k(i), \pi_k(i+1))\}.$

It is easy to observe, that arcs from the set $\mathcal{E}(\Theta)$ connect operations executed on the same machine (π_k is a permutation of operations executed on the machine M_k, that is operations from the set $\mathcal{O}^k$).

Arcs from the set $\mathcal{R}$ determine the operations execution sequence inside jobs (technological order) and arcs from the set $\mathcal{E}(\pi)$ the operations execution sequence on each machine.

Remark 1. *A pair $\Theta = (\mathcal{Q}, \pi(\mathcal{Q})) \in \Phi$ is a feasible solution for the FJSP if and only if $G(\Theta)$ does not includes cycles.*

Let $\Theta = (\mathcal{Q}, \pi(\mathcal{Q})) \in \Phi^{\circ}$ be a feasible solution for the FJSP and let $G(\Theta)$ be a graph connected with it. A sequence of vertexes $(v_1, v_2, \ldots, v_k)$ of the graph $G(\Theta)$ such, that $(v_i, v_{i+1}) \in \mathcal{R} \cup \mathcal{E}(\Theta)$ for $i = 1, 2, \ldots, k-1$, we call a *path* from the vertex v_1 to v_k. By $C(v, u)$ we denote a longest path (called a *critical path*) in the graph $G(\Theta)$ from the vertex v to u ($v, u \in \mathcal{V}$) and by $L(v, u)$ we denote a *length* (sum of vertexes weights) of this path.

It is easy to notice that the time of all operations execution $C_{\max}(\Theta)$ related with the assignment of operations $\mathcal{Q}$ and schedule $\pi(\mathcal{Q})$ equals to the length $L(s, c)$ of the critical path $C(s, c)$ in the graph $G(\Theta)$. A solutions of the FJSP boils down to determining a feasible solution $\Theta = (\mathcal{Q}, \pi(\mathcal{Q})) \in \Phi^{\circ}$ for which the graph connected with this solution $G(\Theta)$ has the shortest critical path, that is it minimizes $L(s, c)$.

If $\Theta = (\mathcal{Q}, \pi(\mathcal{Q})) \in \Phi^{\circ}$ is a feasible solution for the FJSP then $\mathcal{Q} = [\mathcal{Q}^1, \mathcal{Q}^2, \ldots, \mathcal{Q}^q]$, is an assignment of operations to machines and $\pi(\mathcal{Q}) = (\pi_1(\mathcal{Q}), \pi_2(\mathcal{Q}), \ldots, \pi_m(\mathcal{Q}))$ is a concatenation of m permutations, where a permutation $\pi_i(\mathcal{Q})$ determines a sequence of operations from the set $\mathcal{Q}^i$ which have to be executed on the machine M_i ($i = 1, 2, ..., m$).

Let $C(s, c) = (s, v_1, v_2, \ldots, v_w, c)$, $v_i \in \mathcal{O}$ ($1 \leq i \leq w$) be a critical path in the graph $G(\Theta)$ from the starting vertex s to the final vertex c. This path can be divided into subsequences of vertexes $\mathcal{B} = [B^1, B^2, \ldots, B^r]$ called *blocks* in the permutations on the critical path $C(s, c)$ (Grabowski [5]) where

(a) a block is a subsequence of vertexes from the critical path including successive operations executed directly one after other,
(b) a block includes operations executed on the same machine,
(c) a product of any two blocks is an empty set,
(d) a block is a maximal (due to including) subset of operations from the critical path fulfilling constrains (a)-(c).

In the further part only these blocks for which $|B^k| > 1$ will be considered, i.e. non-empty blocks.

If ($k = 1, 2, ..., r$) is a block on the machine M_i ($i = 1, 2, ..., m$) from the type of machines t ($t = 1, 2, ..., q$) then we will denote it as follows:

$$B^k = (\pi_i(a^k), \pi_i(a^{k+1}), \ldots, \pi_i(b^{k-1}), \pi_i(b^k)),$$

where $1 \leq a^k \leq b^k \leq |Q_i^t|$.

Operations $\pi(a^k)$ and $\pi(b^k)$ in the block B^k are called *the first* and *the last*, respectively. In turn a block without the first and the last operation we call an *internal block*.

The change of operations order in any block does not generate the solution with less value of the cost function (see Grabowski [5]). At least one operation from any block should be moved before the first or after the last operation of this block to generate the solution (graph) with smaller weight of the critical path. We use this property to reduce the neighborhood size, i.e. do not generate solutions with greater values (comparing the the current solution) of the cost function.

4 Proposed Algorithm

The algorithm proposed here includes two major modules: the machine selection module and the operation scheduling module.

Machine selection module. This module is based on the tabu search approach and it works sequentially. It helps an operation to select one of the parallel machine from the set of machine types to process it.

Operation scheduling module. This module is used to schedule the sequence and the timing of all operations assigned to each machine from the center. It has to solve classic job shop problems after having assigned operations to machines. Two approaches: constructive INSA [7] and TSAB [7] (tabu search) were used on this level.

On each level a metaheuristic algorithm is used, so we call this method meta^{2}heuristics (*meta-square-heuristics*).

Algorithm 1. Tabu Search Based Meta^{2}heuristics (M^{2}h)

```
$\mathcal{Q}^*$ – the best known assignment;
$\pi(\mathcal{Q}^*)$ – operation sequence corresponding to the best known assignment $\mathcal{Q}*$;
 Step 0. Find start assignment of operations on machines $\mathcal{Q}^0$ and corresponding
         operation sequence $\pi(\mathcal{Q}^0)$;
 Step 1. Generate the neighborhood $\mathcal{N}(\mathcal{Q})$ of the current assignment $\mathcal{Q}$.
         Exclude from $\mathcal{N}(\mathcal{Q})$ elements from tabu list $T$;
 Step 2. Divide $\mathcal{N}(\mathcal{Q})$ into $k = \left\lceil \frac{|\mathcal{N}(\mathcal{Q})|}{p} \right\rceil$ groups;
         Each group consist of at most $p$ elements;
 Step 3. For each group $k$ find (using $p$ processors) operation sequence $\pi(\mathcal{Y})$
         corresponding to the assignment $\mathcal{Y} \in \mathcal{N}(\mathcal{Q})$ and value
         of the makespan $C_{max}(\mathcal{Y}, \pi(\mathcal{Y}))$;
 Step 4. Find assignment $z \in \mathcal{N}(\mathcal{Q})$ such that
              $C_{max}(\mathcal{Z}, \pi(\mathcal{Z})) = min\{C_{max}(\mathcal{Y}, \pi(\mathcal{Y})) : \mathcal{Y} \in \mathcal{N}(x)\}$;
 Step 5. if $C_{max}(\mathcal{Z}, \pi(\mathcal{Z})) < C_{max}(\mathcal{Q}^*, \pi(\mathcal{Q}^*))$ then $\pi(\mathcal{Q}^*) = \pi(\mathcal{Z})$; $\mathcal{Q}^* = \mathcal{Z}$;
         Include $z$ in the list T; $\mathcal{Q} = \mathcal{Z}$; $\pi(\mathcal{Q}) = \pi(\mathcal{Z})$;
 Step 6. if (Stop condition is true) then Stop;
         else go to Step 1;
```

In the second step of the algorithm a neighborhood $\mathcal{N}(\mathcal{Q})$ is divided into disjoint sets $\bigcup_{i=1}^{k} \mathcal{N}_i(\mathcal{Q}) = \mathcal{N}(\mathcal{Q}), \bigcap_{i=1}^{k} \mathcal{N}_i(\mathcal{Q}) = \emptyset$. For each group k values of the makespan are calculated using p GPU processors. Number of processors used in the third step depends on the neighborhood size. In the Step 3 the value of makespan corresponding to the assignment is calculated by means of INSA or TSAB algorithms. Tabu list T stores couples (v, k) where v is the position in the assignment vector and k is the machine to which v is assigned before the move. The first assignment is generated by the search for the global minimum in the processing time table taken from [8].

5 Computational Results

The parallel meta^{2}heuristic (M^{2}h) algorithm for the flexible job shop problem was coded in C (CUDA) for GPU, ran on the Tesla C870 GPU (512 GFLOPS) with 128 streaming processors cores and tested on the benchmark problems from literature. The GPU was installed on the Hewlett-Packard server based on 2 Dual-Core AMD 1 GHz Opteron processors with 1 MB cache memory and 8 GB RAM working under 64-bit Linux Debian 5.0 operating system. We compare our results with results obtained by other authors using a set of 10 problems from Brandimarte [2] and a set of 21 problems from Barnes and Chambers [1].

Table 1. Experimental results of the M^{2}h for Brandimarte [2] instances. The INSA algorithm was used in the operation scheduling module.

problem	$n \times m$	Flex.	o	t_s [s]	t_p [s]	speedup s
Mk01	10×6	2.09	55	133.61	10.79	12.38
Mk02	10×6	4.10	58	218.02	10.55	20.67
Mk03	15×8	3.01	150	6495.35	136.19	47.69
Mk04	15×8	1.91	90	620.69	29.59	20.98
Mk05	15×4	1.71	106	1449.80	74.55	19.45
Mk06	10×15	3.27	150	8094.39	147.83	54.75
Mk07	20×5	2.83	100	1939.33	57.92	33.48
Mk08	20×10	1.43	225	8950.91	643.39	13.91
Mk09	20×10	2.53	240	24586.00	641.88	38.30
Mk10	20×15	2.98	240	31990.55	593.49	53.90

The first phase of computational experiments was devoted to parallelization efficiency determination by estimating experimental speedup values. The sequential algorithm using one GPU processor was coded with the aim of determining the speedup value of the parallel algorithm. Such an approach is called orthodox speedup and it compares times of algorithms execution on machines with the same processors (1 versus p processors). Table 1 shows computational times for the sequential and the parallel algorithm as well as speedup values. The orthodox speedup s value can by set by the following expression $s = \frac{t_s}{t_p}$, where t_s - the computational time of sequential algorithm executed on the single processor of the GPU, t_p - the computational time of parallel algorithm executed on p processors of the GPU. Flex. denotes the average number of equivalent machines per operation. As we can notice the highest speedup values were obtained for the problem instances with a bigger number of jobs n and the number of operations o. In this phase the simple INSA algorithm was applied in the operation scheduling module of the parallel meta^{2}heuristics.

Table 2. Values of the obtaining solutions for Barnes and Chambers [1] instances. The TSAB algorithm was used in the operation scheduling module of the M^{2}h. New the best known solutions are marked out by a bold font.

problem	$n \times m$	(LB,UB)	MG [6]	hGA [4]	M^{2}h
mt10c1	10 × 11	(655,927)	928	927	927
mt10cc	10 × 12	(655,914)	910	910	**908**
mt10x	10 × 11	(655,929)	918	918	922
mt10xx	10 × 12	(655,929)	918	918	918
mt10xxx	10 × 13	(655,936)	918	918	918
mt10xy	10 × 12	(655,913)	906	905	905
mt10xyz	10 × 13	(655,849)	847	849	855
setb4c9	15 × 11	(857,924)	919	914	914
setb4cc	15 × 12	(857,909)	909	914	**907**
setb4x	15 × 11	(846,937)	925	925	925
setb4xx	15 × 12	(846,930)	925	925	925
setb4xxx	15 × 13	(846,925)	925	925	925
setb4xy	15 × 12	(845,924)	916	916	**910**
setb4xyz	15 × 13	(838,914)	905	905	905
seti5c12	15 × 16	(1027,1185)	1174	1175	1174
seti5cc	15 × 17	(955,1136)	1136	1138	1136
seti5x	15 × 16	(955,1218)	1201	1204	**1199**
seti5xx	15 × 17	(955,1204)	1199	1202	**1198**
seti5xxx	15 × 18	(955,1213)	1197	1204	1197
seti5xy	15 × 17	(955,1148)	1136	1136	1136
seti5xyz	15 × 18	(955,1127)	1125	1126	1128

The second phase of the tests was refer to obtaining as good results of the cost function as possible. In this phase specialized TSAB algorithm of Nowicki and Smutnicki [7] was used in the operation scheduling module of the parallel meta^{2}heuristics. Despite of being more time-consuming the quality of the obtained results is much better than in the case of using INSA. By means of this approach it was possible to obtain 5 new the best known solutions for the benchmarks of Barnes and Chambers [1], for instances `mt10cc` (the new value 908), `set64c9` (907), `set64xy` (910), `seti5x` (1199) and `seti5xx` (1198).

The obtained results were also compared to other resent approach from the literature proposed for the flexible job shop problem. The proposed parallel M^{2}h algorithm managed to obtain the average relative percentage deviation to the best known solution of the Barnes and Chambers benchmark instances on the level of 0.014% versus 0.036% of the MG [6] algorithm of Mastrolilli and Gambardella and 0.106% of the hGA [4] algorithm of Gao et al.

6 Conclusions

We have discussed a new approach to the scheduling problems with parallel machines, where assignment of operations to machines defines a classical problem without parallel machines. We propose double-level parallel metaheuristics, where each solution of the higher level, i.e. jobs assignment to machines, defines an NP-hard job shop problem, which we are solving by the second metaheuristics (constructive INSA or tabu search based TSAB) – we call such an approach meta^{2}heuristics. Using exact algorithms on both levels (i.e. branch and bound) makes possible to obtain an optimal solution of the problem.

References

1. Barnes, J.W., Chambers, J.B.: Flexible job shop scheduling by tabu search. Graduate program in operations research snd industrial engineering, The University of Texas at Austin, Technical Report Series: ORP96-09 (1996)
2. Brandimarte, P.: Routing and scheduling in a flexible job shop by tabu search. Annals of Operations Research 41, 157–183 (1993)
3. Dauzère-Pérès, S., Pauli, J.: An integrated approach for modeling and solving the general multiprocessor job shop scheduling problem using tabu search. Annals of Operations Research 70(3), 281–306 (1997)
4. Gao, J., Sun, L., Gen, M.: A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. Computers & Operations Research 35, 2892–2907 (2008)
5. Grabowski, J.: Generalized problems of operations sequencing in the discrete production systems. In: Monographs, vol. 9. Scientific Papers of the Institute of Technical Cybernetics of Wrocław Technical Univercity (1979) (in Polish)
6. Mastrolilli, M., Gambardella, L.M.: Effective neighborhood functions for the flexible job shop problem. Journal of Scheduling 3(1), 3–20 (2000)
7. Nowicki, E., Smutnicki, C.: A fast tabu search algorithm for the permutation flowshop problem. European Journal of Operational Research 91, 160–175 (1996)
8. Pezzella, F., Morganti, G., Ciaschetti, G.: A genetic algorithm for the Flexible Jobschop Scheduling Problem. Computers & Operations Research 35, 3202–3212 (2008)
9. Pinedo, M.: Scheduling: theory, algorithms and systems. Prentice-Hall, Englewood Cliffs (2002)

Particle Swarm Optimization for Container Loading of Nonorthogonal Objects

Isaac Cano and Vicenç Torra

IIIA, Artificial Intelligence Research Institute
CSIC, Spanish National Research Council
cano@iiia.csic.es, vtorra@iiia.csic.es

Abstract. A feasible solution for a container loading problem is the exact order, position and orientation in which the objects are loaded into the container. Several algorithms have been proposed to load objects with different shapes and taking into account a few constraints. The aim of this paper is to provide an approach to solve the problem of loading orthogonal and nonorthogonal boxes, considering them as polyhedral. In addition, our proposal deals with constraints on the dimensions of the container, whether every box can be rotated in any of the three dimensions, the maximum load bearing strength and also the minimum stability required.

1 Introduction

Container loading is a significant area for companies engaged in the transportation of goods and logistics in which the optimization of the resources plays an important role. Container loading problems can be formalized by defining the properties of the container and the objects. Also there is the need to take into account some constraints that determine whether the final solution is acceptable. After these definitions the container loading problem becomes a constrained optimization problem whose function to maximize is the allocated space of the container. Getting an optimal solution is almost unreachable because they are NP-hard problems [5]. Due to this, heuristic methods have to be considered. Those methods are usually specialized by loading the same kind of boxes. This paper generalizes those methods and the generalization will be done by enabling the load of standard boxes, dodecahedron and hexagonal prisms.

This paper extends [4] by using a Particle Swarm Optimization (PSO) algorithm [8] to obtain the weights' values that maximize the average volume occupied. We will assess the performance of our approach when just considering orthogonal boxes by solving the benchmark created by Bischoff and Ratcliff [2]. Later on, we will present the performance of our approach applied to nonorthogonal objects running our proposed benchmark for polyhedral boxes.

The structure of this paper is as follows. In Section 2, we describe step by step the heuristic described in [1] to load standard boxes and its extension for nonorthogonal boxes while giving a detailed overview of the criteria used for box selection. In Section 3, we explain how to solve the problem of getting feasible

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 403–410, 2010.
© Springer-Verlag Berlin Heidelberg 2010

positions for box loading. In Section 4 we present the basics of PSO and later in Section 5 we describe the experiments we have performed and the results obtained.

2 Nonorthogonal Objects: The Heuristic

In this paper we propose an extension of the method proposed by E.E. Bischoff [1] to deal also with nonorthogonal boxes, in particular, dodecahedra and prisms. The container is loaded box by box, applying a heuristic to assess the suitability of the different options for loading a box. Formally, the boxes to be allocated are represented as an array of boxes ordered by volume in decreasing order. Then, at a given point, when a new box is needed, a box is selected on the basis of five criteria. Each criterion is considered for each box, each possible box rotation and each feasible location. We give a description of the heuristic's steps on the following subsections.

2.1 Identify All Fully Supported Spaces

The first step to allocate a box into the container is to identify the available empty spaces which can be considered as candidates for containing the box. When the box can be rotated, different rotations might lead to different sets of possible empty spaces. So, we have to search for the empty spaces of each possible rotation in order to find all feasible box-space combinations.

2.2 Evaluate the Box-Space Combinations Found

E.E. Bischoff in [1] proposed five criteria to evaluate the different box-space combinations. We describe their rationale and our adapted criteria, which take into account that the boxes can be nonorthogonal.

Criterion 1 (+): Relation between box size and position of loading surface. This criterion supports loading big boxes on the bottom so we may have opportunities to load smaller boxes on them. We consider the difference between the height of the container H and the vertical position where the space is located S_H. This difference is then multiplied by the result of the box volume V_B.

$$C_1 = (H - S_H)V_B$$

Criterion 2 (+): Match between box and space dimensions. This second criterion assesses the similarity between a box and the space dimensions. S_A means the amount of available space in this region and S_N denotes the needed space to load on this region the box in process.

$$C_2 = \frac{S_N}{S_A}$$

Criterion 3 (-): Unusable space generated. In some sense, this criterion is the opposite of the previous one but taking into account if the type of the box in process is a prism, a dodecahedron or a simple orthogonal box. Hence this criterion will be weighted with a greater value in case of dodecahedron ($\alpha = 1.4$) because of the fact that this kind of box generates more unusable spaces than a prism or an orthogonal box. In the same way, in the case of prisms ($\alpha = 1.2$) and in the case of orthogonal boxes ($\alpha = 1$). The weights for the different box types have been obtained heuristically.

$$C_3 = \alpha - (S_A - S_N)$$

Criterion 4 (+): Potential for building column of identical boxes. This criterion supports the cases that maximize the similarity between box and space dimensions and minimize the unusable space generated while remaining boxes of the same type. In this formula m_i means the number of boxes of the same type as the actual one that remains unloaded. d_{i1}, d_{i2} and d_{i3} represents the box's length, width and height. S_L and S_w takes the value of the horizontal position along the container length and width. It is also considered the minimum load bearing capacity on the top surface B_{top} on which the actual box will be placed.

$$C_4 = min\{m_i, \tfrac{H-S_H}{d_{i3}}, 1 + \frac{B_{top} d_{i1} d_{i2}}{w_i}\} \times \frac{V_B}{S_L S_w}$$

$$B_{top} = min\{b_{i3}, B_{min} - \frac{w_i}{d_{i1} d_{i2}}\}$$

Criterion 5 (-): Relative loss in load bearing capacity. A big value of this criterion means that we are trying to load a heavy box on the top of another one decreasing its load bearing capacity and causing a strong impact on subsequent placement opportunities. This criterion rewards loading heavy boxes in low altitudes. The best reward is given when heavy boxes are on ground levels.

$$C_5 = (B_{avg} - B_{top} d_{i1} d_{i2}) \times \tfrac{H-S_H-d_{i3}}{d_{i3}}$$

This formula will be applied only if C_5 meets the following inequation:

$$\tfrac{B_{top}}{H-S_H-d_{i3}} < D_{max}$$

Otherwise $C_5 = 0$. Hence with this criterion we only penalize when after loading the box, we surely know that the box with largest density cannot be placed in the future above the loaded box.

2.3 Select the Combination with the Highest Score and Place the Box

Once we have the value of the five criteria for each feasible box-space combination, the overall score for each combination is calculated as a weighted sum of the five criteria. Details are given in Section 3.

3 Nonorthogonal Objects: Solving the Problem

In this Section we describe in more detail some of the steps described above. In particular, we discuss how to identify all fully supported spaces, how to handle the stability and load bearing strength constraints and how to calculate the overall score for every box-space combination to select the one with highest score to load the current box. Also, we will describe how we ensure the stability and maximum load bearing strength compliance.

3.1 Getting Potential Spaces Where to Load the Box

The method proposed to get the box-space combinations uses a 2D matrix, $length \times width$ of the container, containing the up to date height of all boxes already loaded. Each space found is described by its area and starting position (upper left corner) X, Y and Z. The area is used to filter the spaces according to the free space needed to load the box.

The method gets all feasible spaces looking on the entire 2D matrix for spaces of the same height. The starting position of the space is a vertex. Formally, the vertex management approach [9] uses a list of vertexes where objects can be loaded. The first vertex is $(0, 0, 0)$ and after a box is loaded the corresponding vertex is removed from the list adding some new vertexes. For all box's types three new vertexes are added, $(x_i + w_i, y_i, z_i)$, $(x_i, y_i + d_i, z_i)$ and $(x_i, y_i, z_i + h_i)$, where (x_i, y_i, z_i) states for the reference position of the box b_i and w_i, d_i, h_i represents its dimensions. In case of a nonorthogonal box some heuristically selected vertexes are added too in order to allow the possibility to load small boxes below a dodecahedron or to form honeycomb-like structures with prisms. When a space is found, it is marked as visited and the amount of marked coordinates is defined as its area.

The procedure to maintain the heights up to date in the 2D matrix is applied after every box b_i is loaded. The input of this procedure is the box's position, length and width. The output of the procedure is the height for all coordinates (i, j) for $i = x_i$ to $x_i + d_i$ and $j = y_i$ to $y_i + w_i$.

3.2 Handling the Load Bearing Strength Constraint

The method proposed to support the maximum load bearing strength associated to each dimension of a box is based in another 2D data structure similar to the one used to get the potential spaces where to load the box. The same procedure is applied after every single box is loaded inside the container and it takes into account the remaining load bearing strength of the space before the box was loaded, the box's weight and the box's load bearing strength. Hence, the input of this procedure is the box's position, length and width and also the box's load bearing capacity and weight. The output of the procedure is the remaining load bearing strength after the box has been loaded or the box's load bearing strength in case the former is bigger than the latter.

3.3 Handling the Stability Constraint

To ensure the compliance of the stability constraint, we use the two dimensional data structure used to get potential spaces where to load the box. This is an easy process where we have to ensure that the height values corresponding to the space where to load the box are most of them equal to the box's starting at coordinate z_i. The percentage of equal values can be used to ensure a degree of stability. This is to say, if we ensure a 100% stability compliance all the height values corresponding to the space where to load the box have to be equal to z_i.

3.4 Criteria: Overall Score

Taking into account the criteria above, the following score is computed for each box: $E(box) = v_1C'_1(box) + v_2C'_2(box) + v_3C'_3(box) + v_4C'_4(box) + v_5C'_5(box)$. In this definition, we have assumed that all criteria, including criteria 3 and 5, are positive and that their values are bounded (they belong to the unit interval). In addition, we require the weights v_i add to one (i.e., $\sum_{i=1}^{5} v_i = 1$). In this way, the expression is a weighted mean of the criteria. Formally, $C'_i = C_i/r_i$ where $r_i = max_{k=1}^{n} r_k$ for $i = 1, 2$, and 4, and $C'_i = C_i/r_i$ where $r_i = min_{k=1}^{n} r_k$ for $i = 3$ and 5. r_i stands for the maximum/minimum value of the i-*th* criterion among all the box-space combinations assessed.

4 Particle Swarm Optimization (PSO)

Since 1995 when James Kennedy and Eberhart proposed the Particle Swarm Optimization algorithm [8], some extensions and optimizations of their parameters have been realized [11]. PSO is a population based stochastic optimization technique inspired by the social behavior of bird flocking or fish schooling. PSO shares many similarities with Genetic Algorithms (GA), the system is randomly initialized and searches for optima by updating generations. However, unlike GA, PSO has no evolution operations such as crossover or mutation. In PSO, the potential solutions are called particles and they fly through the problem space by following the optimum particles. Compared to GA, the advantages of PSO are that PSO is easy to implement and there are few parameters to adjust.

```
foreach particle do
    Randomly initialize particle
end
repeat
    foreach particle do
        Calculate fitness value;
        if the fitness value is better than the best fitness value in history then
            Set current fitness value as the best fitness value;
        end
    end
    Choose the particle with the best fitness value of all the particles;
    foreach particle do
        Calculate particle's velocity according to equation (1);
        Calculate particle's position according to equation (2);
    end
until maximum iterations or minimum error criteria is attained ;
```

Algorithm 1. Pseudocode of the basic PSO algorithm.

In PSO, each particle has a position $\boldsymbol{p}_i$ and a velocity $\boldsymbol{v}_i$. Once the particles are randomly distributed in the search space the position and velocity of each particle are modified by combining some aspect of its experience, as its best position found $\boldsymbol{b}_i$, with social information, as the best position of its neighbors $\boldsymbol{g}_i$. At each algorithm iteration, particles evaluate the objective function (fitness value) $f_u(\boldsymbol{p}_i)$ at its current location.

The movement of each particle follows the next two equations: (1) $\boldsymbol{v}_i = \chi(\boldsymbol{v}_i + \boldsymbol{U}(0, \phi_1)(\boldsymbol{b}_i - \boldsymbol{p}_i) + \boldsymbol{U}(0, \phi_2)(\boldsymbol{g}_i - \boldsymbol{p}_i))$ and (2) $\boldsymbol{p}_i = \boldsymbol{p}_i + \boldsymbol{v}_i$. Where χ is the constant multiplier that ensures the convergence, $\boldsymbol{p}_i$ is the current position of the particle i, $\boldsymbol{v}_i$ is the velocity of the particle i, $\boldsymbol{b}_i$ is the best position found by the particle i and $\boldsymbol{U}(0, \phi_i)$ represents a vector of random numbers uniformly distributed in $[0, \phi_i]$. The particles' velocity is updated by means of equation (1), which is composed by the cognitive part $\boldsymbol{U}(0, \phi_1)(\boldsymbol{b}_i - \boldsymbol{p}_i)$ and the social part $\boldsymbol{U}(0, \phi_2)(\boldsymbol{g}_i - \boldsymbol{p}_i)$. The basic PSO algorithm is shown in algorithm 1.

Using PSO to find out the best weights $\boldsymbol{v_i}$

The above score is used for selecting which of the available boxes should be loaded next in the container. Nevertheless, the score depends on the weights v_i. We have applied the PSO procedure to define these weights. The standard PSO parameters have been settled following [3], $\chi = 0.729843788$, $\phi_1 = \phi_2 = 2.05$. Note that the goal is to obtain weights that result in a good average volume of occupation for all the problems in the benchmarks. The results obtained are analyzed in Section 4.

5 Experiments

In this section we describe the experiments performed and analyze the results. Two sets of experiments have been considered. The weights used in the experiments were obtained by means of PSO and their values are the following: $w1 = [0.396755737, 0.0, 0.284134413, 0.240106459, 0.079003391]$.

Bischoff and Ratcliff files

To compare our approach to previous work dealing only with orthogonal boxes we have considered the problems described in [2] and accessible through the web site in [14]. They consist of 7 sets of 100 problems. In each set, all problems have the same number of different boxes, although the sizes of the boxes are not the same. Test files can be found in [15].

Table 1 shows that the results obtained when just considering the stability constraint (LBA OFF) are slightly worse than the ones obtained by Bischoff and Ratcliff. A difference of about 4% or less is obtained. Nevertheless, difference is at the cost of permitting additional shapes for the boxes. In any case, for some of the particular problems we achieve a good score e.g. 88.51% or 88.20%.

To assess the impact of the load bearing ability constraint on the container loading process, Table 1 shows the results when considering five different degrees of load bearing ability LBA. Starting from a degree of load bearing strength,

Table 1. Average and maximum occupation, standard deviation and average execution time when processing the Bischoff&Ratcliff files without considering the load bearing ability (LBA OFF) and decreasing degrees of load bearing ability (LBA TEST 1…5)

		wtpack1	wtpack2	wtpack3	wtpack4	wtpack5	wtpack6	wtpack7
Avg. Occup. (%)	LBA OFF	76.18	76.18	75.88	76.07	76.09	76.17	76.09
	LBA TEST 1	70.22	70.27	69.91	70.64	69.87	70.81	70.50
	LBA TEST 2	69.16	69.02	68.90	68.85	68.91	69.04	69.01
	LBA TEST 3	67.55	67.50	67.74	68.24	67.70	67.51	67.94
Max. Occup. (%)	LBA OFF	88.20	88.19	87.47	87.34	88.51	87.51	87.87
	LBA TEST 1	88.19	86.94	87.20	87.47	87.47	87.20	86.23
	LBA TEST 2	87.87	86.94	87.87	87.47	88.19	87.60	86.76
	LBA TEST 3	86.77	88.19	86.21	88.19	86.21	88.19	86.52
St. Dev. (%)	LBA OFF	06.25	06.25	06.12	06.12	06.30	06.21	05.97
	LBA TEST 1	12.25	12.48	12.38	12.11	12.44	12.15	12.49
	LBA TEST 2	13.15	13.09	13.21	13.08	13.25	13.04	13.29
	LBA TEST 3	13.40	13.56	13.25	13.56	13.47	13.95	13.72
Avg. Time (sec.)	LBA OFF	40.11	40.09	40.21	40.80	39.22	41.19	39.51
	LBA TEST 1	27.11	26.00	29.96	28.49	27.61	32.36	30.87
	LBA TEST 2	27.79	26.43	25.33	24.68	23.78	27.09	25.04
	LBA TEST 3	22.68	21.74	24.28	24.48	21.80	22.87	23.39

(LBA TEST 1), and decreasing the load bearing strength down to (LBA TEST 5). The results are slightly better (1%) than the previous work [4] for the case of LBA OFF but when considering the different degrees of LBA we improved the results from 7% up to 10%.

Variation of Bischoff and Ratcliff files

This is a variation of the files described in the previous section, where other kinds of boxes are also permitted. We have considered, as in the Bischoff and Ratcliff files, sets of 3, 5, 8, 10, 12, 15 and 20 different box types. Then, for each of these sets of boxes, we have considered different cases corresponding to different proportions of boxes of different shapes. Table 2 gives the 7 cases considered. That is, we have considered the case of only boxes (as in Bischoff and Ratcliff files), the case of 50% regular orthogonal boxes and 50% hexagonal prisms, the case of 50% regular orthogonal boxes and 50% dodecahedron, etc. The data files are publicly available through the web page given in [15].

Table 2. Results of our variation of Bischoff&Ratcliff files considering load bearing strength, and mean occupancy according to box types, method used

Case	Boxes	Dodec.	Hex. prism	Avg. Occup.	Max. Occup.	St. Dev.	Avg. Time
Case 1	100%						
Case 2	50%		50%	66.67	87.56	05.88	13.23
Case 3	50%	50%		74.05	96.25	11.03	17.42
Case 4			100%	68.77	88.28	07.95	157.04
Case 5		50%	50%	63.76	78.95	09.50	83.60
Case 6		100%		62.08	69.06	06.05	29.73
Case 7	33%	33%	33%	70.01	82.57	07.63	40.65

The results obtained are better than the ones obtained in the previous work [4]. In average, we have improved the results in 18%. Nevertheless, dodecahedra continue representing the worst case.

6 Conclusions

In this paper we have considered the container loading problem for nonorthogonal objects. Our approach has been applied to three types of objects: typical boxes (orthogonal boxes), dodecahedra and hexagonal prisms. We have presented the results of our approach that are similar to the ones described in Bischoff and Ratcliff but permitting its application to the new types of objects and considering both the stability and maximum load bearing ability constraints.

Acknowledgments

This work is partially supported by the Spanish Ministerio de Fomento (project PON, T27/2006) and MEC (CONSOLIDER INGENIO 2010 CSD2007-00004, and TSI2007-65406-C03-02).

References

1. Bischoff, E.E.: Three-dimensional packing of items with limited load bearing strength. European Journal of Operational Research 168, 952–966 (2006)
2. Bischoff, E.E., Ratcliff, M.S.W.: Issues in the Development of Approaches to Container Loading. Omega Int. J. of Management Sciences 23(4), 377–390 (1995)
3. Blackwell, T.: Particle swarm optimization in dynamic environments. In: Yang, S., Ong, Y.-S., Jin, Y. (eds.) Evolutionary Computation in Dynamic and Uncertain Environments, pp. 29–49 (2007)
4. Cano, I., Torra, V.: Container loading for nonorthogonal objects with stability and load bearing strength compliance. In: Proceedings of LINDI 2009 (2009)
5. Eley, M.: Solving container loading problems by block arrangement. European Journal of Operational Research 141, 393–409 (2002)
6. Gehring, H., Bortfeldt, A.: A Genetic Algorithm for Solving the Container Loading Problem. Int. Trans. Operational Research 4(5/6), 401–418 (1997)
7. George, J.A., George, J.M., Lamar, B.W.: Packing different-sized circles into a rectangular container. European Journal of Operational Research 84, 693–712 (1995)
8. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. IEEE International Conference on Neural Networks, vol. 5, pp. 1942–1948 (1995)
9. Miyamoto, S., Endo, Y., Hanzawa, K., Hamasuna, Y.: Metaheuristic Algorithms for Container Loading Problems: Framework and Knowledge Utilization. J. of Advanced Intelligence and Intelligent Informatics 11(5), 51–60 (2006)
10. Pisinger, D.: Heuristics for the container loading problem. European Journal of Operational Research 141, 382–392 (2002)
11. Poli, R., Kennedy, J., Blackwell, T.: Particle swarm optimization. An overview. Swarm Intelligence 1, 33–57 (2007)
12. Torra, V., Cano, I., Miyamoto, S., Endo, Y.: Container loading for nonorthogonal objects: an approximation using local search and simulated annealing. In: Soft Computing - A Fusion of Foundations, Methodologies and Applications, pp. 537–544 (2009)
13. Zachmann, G.: Exact and Fast Collision Detection, Diploma Thesis, Technischen Universität Darmstadt (1994)
14. http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/wtpack1.txt (to wtpack7.txt)
15. http://www.iiia.csic.es/~vtorra/projecte.pon/

Distributed Control of Illuminance and Color Temperature in Intelligent Lighting System

Chitose Tomishima[1], Mitsunori Miki[2], Maiko Ashibe[1],
Tomoyuki Hiroyasu[3], and Masato Yoshimi[2]

[1] Graduate School of Engineering, Doshisha University
[2] Depertment of Science and Engineering, Doshisha University
[3] Department of Life and Medical Sciences, Doshisha University
1-3 Tatara Miyakodani, Kyotanabe, Kyoto 610-0321, Japan
{ctomishima@mikilab,mmiki@mail}.doshisha.ac.jp, asshi219@gmail.com,
{tomo@is,myoshimi@mikilab}.doshisha.ac.jp

Abstract. This reserch proposes a method to provide an appropriate illuminance and color temperature in an office lighting environment for individual workers. For the control algorithm, based on the optimization method called Simulated Annealing (SA), a method including a neighborhood design mechanism for the design variables instead of a temperature parameter was used. In addition, for changes in the luminous intensity of high color temperature light and low color temperature light, five neighborhoods were adaptively used. Through the experiment using the daylight color fluorescent lamps and warm white color fluorescent lamps, the illuminance and color temperature converged into the target values.

Keywords: intelligent lighting system, autonomous distributed control, energy saving, Illuminance, Color Temperature.

1 Introduction

Recently, the enhancement of the intellectual productivity, creativity and comfort of office workers are being focused on. A great deal of research into the effects of office environments on intellectual productivity has already been conducted. This research has reported that intellectual productivity is enhanced by improving the office environment [1]. Research focusing on lighting environments in offices has reported that changes in illuminance and color temperature corresponding to biological rhythm enhance intellectual productivity [1]. There is also research reporting that individual workers need different illuminance depending on job description [2]. With this background, the authors researched an intelligent lighting system that provides different levels of illuminance for individual office workers [3]. The intelligent lighting system is a distributed autonomous lighting system that provides an appropriate illuminance to an appropriate location. Such intelligent lighting system have already been tested in actual offices and their high performance and practical utility have been verified.

The intelligent lighting system focuses on illuminance that is an element of the light environment. Illuminance is the brightness of any given place. Color

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 411–419, 2010.
© Springer-Verlag Berlin Heidelberg 2010

temperature is also important for lighting environments, however. This research focuses on illuminance and color temperature and proposes the individually distributed control of illuminance and color temperature. By providing different levels of illuminance and color temperature to office workers, a light environment that depends on individual conditions can be realized. Through this, it is expected that intellectual productivity, creativity and comfort will be further enhanced. Color temperature will be explained in detail in the following section.

2 Color Temperature

Color temperature is an index that shows the color of light. If a black body is heated, light is eradiated. The temperature of the black body is the color temperature of the light. K (Kelvin) is used for the unit. Lower color temperatures give off a reddish color, while higher color temperatures give off a bluish color. Fluorescent lamps are classified into gdaylight color (about 5000 to 6500 K)h, gcool white color (about 4100 to 4500 K)h, gwarm white color (about 2700 to 3500 K)h from several reteratures[4,5,6,7]. In Japan, the fluorescent lamps with color temperatures of 5000 K are used in common offices.

Various research efforts on the effects of color temperature on humans have been conducted. Research about the effects of color temperature on arousal has reported that light sources with higher color temperatures increase arousal more than lower color temperatures [8]. In addition, higher color temperatures accelerate the excitement of the autonomic nervous system more than lower color temperatures [9]. Research has been conducted on color temperatures appropriate for human behavior, reporting that it is important to provide color temperatures appropriate for living conditions (the design of the lighting environment) [1]. The effects of higher color temperatures on humans have also been researched. Philips conducted an experiment using lights with color temperatures of 17000 K (Acti-Viva fluorescent lamps) at factories and offices. This research showed high color temperature light enhanced work efficiency (e.g., memory was increased, concentration was enhanced and discretion was improved) [10]. Kruithof reported that a lower illuminance was comfortable in lower color temperatures, while a higher illuminance was comfortable in higher color temperatures [11].

As indicated above, it is believed that individual office workers require different levels of illuminance and color temperature depending on the content of their work, conditions and moods. This research proposes an intelligent lighting system that allows for individually distributed control of illuminance and color temperature in an office.

3 Proposed Intelligent Lighting System

3.1 Outline of the Previous System and New System

In the intelligent lighting system, multiple luminaires are connected to the network. Each luminaire satisfies illuminance requested by a user using a built-in microprocessor and a distributed autonomous system or algorithm [3]. The appropriate illuminance is provided to the appropriate location simply by the user setting the target illuminance on the illuminance sensor, without using the position information from the lights and sensors. In addition, the power usage is reduced.

In conventional intelligent lighting system, illuminance is controlled by a distributed control method. This research focuses on illuminance and color temperature and proposes a distributed control of illuminance and color temperature. In this study, as shown in Fig. 1, a chroma sensor and two lights with different color temperatures, i.e., a higher color temperature light and a lower color temperature light, provide individually distributed control illuminance and color temperature. Different levels of illuminance and color temperature are provided in different places by individually distributed control of these lights.

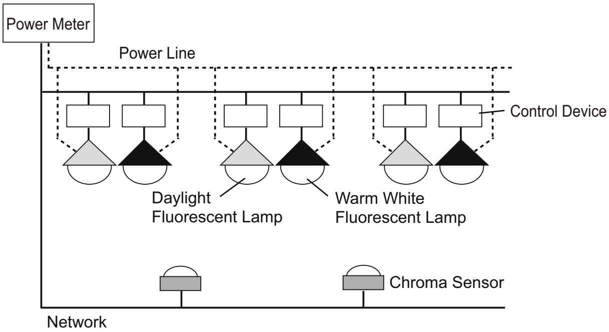


Fig. 1. Configuration of the intelligent lighting system

3.2 Control Algorithm

In the proposed system, lights containing controllers adjust luminous intensity autonomously through a distributed autonomous control algorithm. Luminous intensity is the brightness of a light source. This algorithm is the Adaptive Neighborhood Algorithm using Correlation Coefficient (ANA/CC) [3] containing a neighborhood design mechanism variable instead of a temperature parameter based on Simulated Annealing (SA).

The purpose of this system is to minimize power usage while satisfying the target illuminance and target color temperature set in each sensor. The question to satisfy the purpose of this system is understood as an optimization problem. The purpose of this system is formulated to meet the objective function. The objective function value is maintained by each light independently. Each light minimizes its objective function to optimize the entire system. Equation 1 shows the objective function.

$$f = P + w_1 \sum_{j=1}^{n} L_j + w_2 \sum_{j=1}^{n} T_j \tag{1}$$

$$P = \sum_{i=1}^{m} Cd_i$$

$$L_j = \begin{cases} R_j(Lc_j - Lt_j)^2, & 50 < (Lc_j - Lt_j) \\ 0, & -50 \leq (Lc_j - Lt_j) \leq 50 \\ R_j(Lc_j - Lt_j)^2, & (Lc_j - Lt_j) < -50 \end{cases}$$

$$T_j = \begin{cases} R_j(Tc_j - Tt_j)^2, & 50 < (Tc_j - Tt_j) \\ 0, & -50 \leq (Tc_j - Tt_j) \leq 50 \\ R_j(Tc_j - Tt_j)^2, & (Tc_j - Tt_j) < -50 \end{cases}$$

$$R_j = \begin{cases} r_j, & r_j \geq Threshold \\ 0, & r_j < Threshold \end{cases}$$

n: *number of chroma sensors*, m: *number of lighting fixtures*, w_1, w_2: *weight*
P : *electricity usage amount*, Lc : *current illuminance*, Lt : *target illuminance*
Cd:*luminous intensity*, Tc:*current color temperature*, Tt:*target color temperature*
r : *correlation coefficient*, $Threshold$: *threshold value*

The objective function value is the sum of electricity, illuminance restriction and color temperature restriction. The convergence range of illuminance and color temperature is set to target value ± 50. In other cases, the objective function value is increased as a penalty. Illuminance restrictions and color temperature restrictions are obtained by multiplying the square of the difference between the target value and the current value by the correlation coefficient concerning luminous intensity variation and illuminance variation. If the correlation coefficient is less than the set threshold, it is multiplied by 0. In other words, even if the chroma sensor does not satisfy the target, when the correlation coefficient to the chroma sensor is low, no penalty is given to the objective function. As a result, we can narrow the target of optimization to chroma sensors with higher correlation or chroma sensors with a higher degree of incidence to improve the accuracy to satisfy the target illuminance, target color temperature and convergence time. In addition, illuminance restrictions and color temperature restrictions are multiplied by weight w_1 and w_2 to make it possible to determine the preference among illuminance convergence, color temperature convergence and minimization of electricity based on the value of the weight. The flow of the control algorithm is shown below:

Step1: All lights are turned on with the initial luminous intensity.
Step2: Obtain sensor information from each chroma sensor (i.e., sensor ID, current illuminance, target illuminance, current color temperature and target color temperature) and electricity usage amount from power meter. Target function value is calculated from these values.
Step3: Each light selects one neighborhood from among the neighborhood set by the correlation coefficient described later.
Step4: The next luminous intensity is randomly generated for the neighborhood determined in Step 3. Lights are turned on with the next luminous intensity.
Step5: Obtain sensor information from each chroma sensor and electricity usage amount from power meter.
Step6: Calculate the objective function value from the sensor information and electricity usage amount in the state the lights are turned on with the next luminous intensity.

Step7: Calculate the correlation coefficient from the obtained variation in the illuminance from the chroma sensors and the variation in the luminous intensity of the lights.

Step8: If the objective function value is favorable, the luminous intensity is fixed and the process returns to Step 2. If the objective function value changes for the worse, the changed luminous intensity is cancelled on the basis of the calculation and the process returns to Step 2.

By this algorithm, the degree of incidence of the lights and the chroma sensors is determined, the target illuminance and target color temperature are satisfied and the electricity usage amount is saved in a short time. The reason the process returns to Step 2 instead of Step 3 in Step 8 is to adapt to changes in the environment such as the transfer of chroma sensors and the incidence of outside light.

3.3 The Correlation Coefficient

It is effective to understand the positional relationship between a light and a sensor in order to satisfy the target illuminance and target color temperature and realize an electrical power saving state in a short time. It is efficient to use the correlative relationship between the variation in the luminous intensity of the lights and the variation in the illuminance from the sensors to understand the positional relationship in an autonomous manner. The chart in Fig.2(b) shows the history of the correlation coefficient in the positional relationship between the lights and sensors as shown in Fig.2(a). The correlation between the variation in the luminous intensity of the Light1 and the variation in the illuminance from the sensors becomes higher, and the correlations between the variation in the luminous intensity of the Light2 and 3 and the variation in the illuminance from the sensors becomes lower. By putting this correlation coefficient in equation 1 and using it when generating the next luminous intensity, it is possible to shorten the time to realize the optimum lighting pattern.

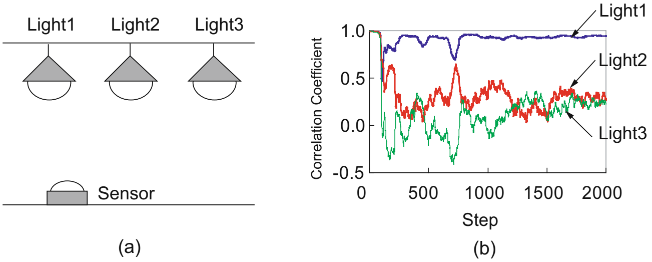


Fig. 2. Correlation coefficient between the lights and the sensors

3.4 Neighborhood Design

Neighborhood is the variation range of the design variable. Neighborhood design refers to the design of the variation range in an appropriate manner. The five neighborhoods shown in Fig.3 are used to randomly increase and decrease the light of the lights. Type A is the neighborhood focusing on decreasing luminous intensity rapidly. Type E is the neighborhood that increases luminous intensity rapidly. Type C is the neighborhood that adjusts luminous intensity when illuminance and color temperature constraint conditions are satisfied. Types B and D are neighborhoods that have intermediate characteristics. The value in Fig.3 is the ratio of the total luminous intensity of the lights (100 % luminous intensity) and the best value obtained in the experiment.

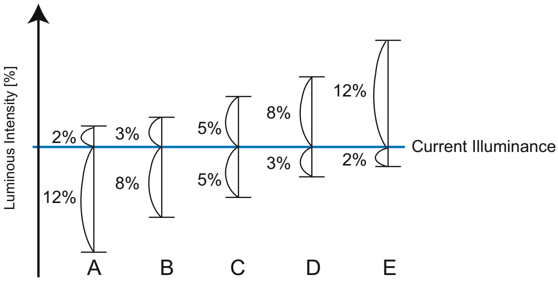
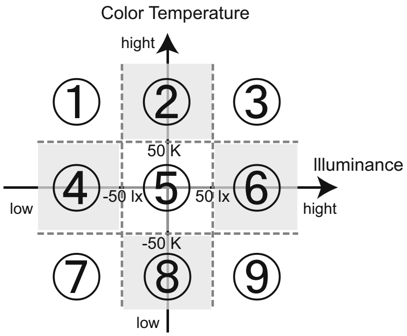


Fig. 3. Five types of the neighborhood rangesivariation range of the design variablej

Based on the illuminance information and color temperature information, the appropriate neighborhood is selected from the five neighborhoods in accordance with neighborhood design. Fig.4 shows the rules of neighborhood design used in this algorithm. The origin in Fig.4 is the target illuminance and target color temperature given to the sensor. As mentioned above, target illuminance ± 50 lx and target color temperature ± 50 K is the convergence range. An illuminance variation of ± 50 lx cannot be detected by humans [12]. In addition, there is little research about variations in color temperature that can be detected by humans. In this research, it was set to ± 50 K based on a preliminary experiment. In conventional intelligent lighting system realizing individual illuminance, the only constraint conditions are the multiple levels of illuminance. If the luminous intensity is increased, the electricity usage amount is also increased, but the restriction on the illuminance level was satisfied. When both the illuminance and color temperature are controlled closer to the target value, however, it is difficult to adjust luminous intensity. In higher color temperature lights, if the luminous intensity is increased, illuminance and color temperature also increase. Meanwhile, in lower color temperature lights, if the luminous intensity is increased, illuminance increases while color temperature decreases. Therefore, for illuminance and color temperature, there are three states, i.e., convergent to target value, lower than target value and higher than target value. As shown in Fig.4 there are nine states in total. In each region, it is possible to converge into the target illuminance and target color temperature by determining the neighborhood used by higher color temperature lights and lower temperature lights. For example, in the range of type A, the illuminance is low and the color temperature is high. At this time, illuminance is increased and color temperature is decreased by increasing the luminous intensity of the lower color temperature light. Accordingly, the neighborhood of the higher color temperature light that does not need to be changed is type C in Fig.4. The neighborhood of the lower color temperature light for which the luminous intensity should be increased is type E in Fig.4. The table in Fig.4 shows the combination of light neighborhoods in a total of nine regions.



Region	Neighborhood	
	Neutral color Fluorescent Lamp	Warm White Fluorescent Lamp
①	C	E
②	B	D
③	A	C
④	D	D
⑤	C	C
⑥	B	B
⑦	E	C
⑧	D	B
⑨	C	A

Fig. 4. Determination method of the neighborhood designs

As shown above, rapid convergence is accelerated by considering the characteristics of each light and the illuminance and color temperature measured by each sensor and generating the next luminous intensity of each light. If a light affects multiple sensors, neighborhood is determined by the illuminance and color temperature information from the sensor with a higher correlation coefficient (i.e., the nearest sensor).

4 Validation Experiment

4.1 Outline of Experiment

An intelligent lighting system that controls illuminance and color temperature in an individual and distributed manner is constructed to verify its effectiveness. Fig.5 shows the experiment layout seen from above. The lights used are daylight color fluorescent lamps (4900 K) and warm white color fluorescent lamps (2800 K) with dimmer controls. For target values, sensor A is 950 lx and 3900 K and sensor B is 1050 lx and 3300 K. The number of searches was 400 and the number of trials was five.

4.2 Result of the Experiment

Fig.6 shows the illuminance history. Fig.7 shows the color temperature history. Based on the results of the experiment, the illuminance from the two sensors converged into the target value after approximately 50 steps. The color temperature converged into the target value after approximately 70 searches. Based on

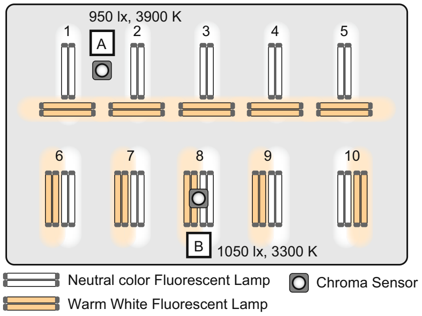


Fig. 5. Experimentation environment

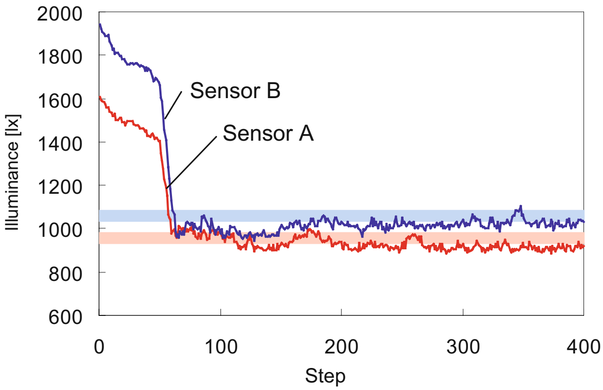


Fig. 6. Result of illuminance

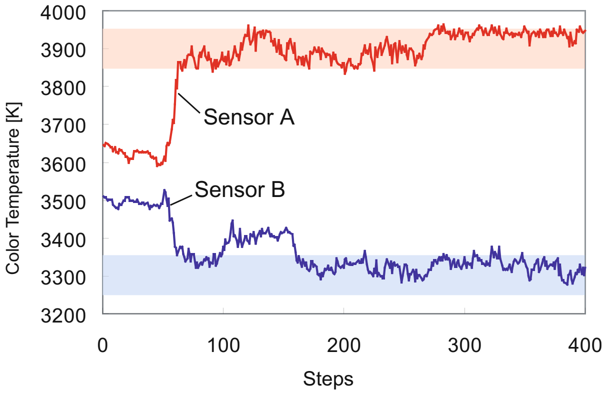


Fig. 7. Result of color temperature

these results, the given illuminance and color temperature were controlled in an individual and distributed manner for a given area using the proposed system.

5 Conclusion

In this report, an intelligent lighting system that provides individually distributed control of illuminance and color temperature is proposed and constructed. For the optimization algorithm, the effect on the sensors is considered and a method using a neighborhood rule categorizing illuminance and color temperature is proposed. The operational experiment shows that the proposed algorithm is effective.

References

1. Obayashi, F., Kawauchi, M., Terano, M., Tomita, K., Hattori, Y., Shimoda, H., Ishii, H., Yoshikawa, H.: Development of an illumination control method to improve office productivity. In: 12th International Conference on Human-Computer Interaction, vol. 9(2), pp. 939–947 (2007)
2. Peter, R.: Boyce.: Individual lighting control: Task, performance, mood and illuminance. Journal of the Illuminating Engineering Society (2000)
3. Miki, M., Hiroyasu, T., Imazato, K.: Proposal for an intelligent lighting system, and verification of control method effectiveness. In: Proc. IEEE CIS, pp. 520–525 (2004)
4. Philips. How to choose the right fluorescent lamp, http://www.lighting.philips.com/us_en/products/homelight/choose.php?main=us_en_consumer_lighting&parent=7593748565&id=us_en_products&lang=en
5. CSSnorthamerica. Lamp comparison chart, http://www.cssnorthamerica.com/hi-spectrum.htm
6. LEPAC. Lighting Guide, http://www.iar.unicamp.br/lab/
7. OEE. Lighting Reference Guide? Fluorescent Lamps, http://oee.nrcan.gc.ca/publications/equipment/lighting/section7.cfm?attr=0
8. Deguchi, T., Sato, M.: The effect of color temperature of lighting sources on mental activity level. The Annals of physiological anthropology 11(1), 37–43, 19920101
9. Mukae, H., Sato, M.: The effect of color temperature of lighting sources on the autonomic nervous functions. The Annals of Physiological Anthropology 11(5), 533–538 (1992)
10. Philips. ActiViva club, http://www.lighting.philips.com/glen/activiva
11. Kruithof, A.A.: Tubular luminescence lamps for general illumination. Philips Tech. Review 6, 65–96 (1941)
12. Shikakura, T., Morujawa, H., Nakamura, Y.: Perception of lighting fluctuation in office lighting environment. Journal of light and visual environment 27(2), 75–82, 20030800

Adaptive Spring Systems for Shape Programming

Maja Czoków and Tomasz Schreiber

Faculty of Mathematics and Computer Science,
Nicolaus Copernicus University,
Toruń, Poland
{maja123,tomeks}@mat.umk.pl

Abstract. We develop a learning algorithm for complex spring networks, aimed at adjusting their physical parameters so as to ensure a desired mechanical behaviour in response to physical input (control) stimuli. The algorithm is based on the gradient descent paradigm and has been tested on our computer implementation. The systems output by our software conform to real-world physics and thus are also suitable for hardware implementation.

1 Introduction

In this paper, we advocate use of complex spring networks as adaptive systems for *shape programming*. The principal task for such systems, embedded in a physical space (usually $\mathbb{R}^2$ in this paper), is to assume a required shape (physical locations of network nodes, regarded as output) in response to suitable physical stimuli (displacements of control nodes, regarded as input). We show how such systems can be implemented and, most importantly, we develop a suitable learning algorithm where the physical spring parameters (rest length, elastic constants) are iteratively adjusted to accomplish the shape programming task specified by a collection of training examples. Our algorithm, of a gradient-descent type, exhibits many common features with classical methods of supervised learning for artificial neural networks (e.g. backpropagation). Its objective may be regarded as designing a mechanical spring system whose behaviour reproduces the desired relationship between displacements of control units and effectors. The applied gradient descent paradigm does guarantee a reliable performance of the developed system.

The spring systems are widely used for modeling large scale elastic properties of physical systems [3,6], disordered media in material sciences [4,10], self-organisation and system design in material and architectural sciences [5,8] and in many other contexts. Moreover, under many aspects the study of spring systems can also be regarded as a particular instance of finite element methods for partial differential equations, and it is beyond the scope of this paper to discuss the immense literature on this classical subject, see again [3]. However, to the best of our knowledge our paper is the first one to propose the use of adaptive spring systems as learning mechanical devices capable of assuming desired physical shapes in response to input displacements of control units. This approach is

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 420–427, 2010.
© Springer-Verlag Berlin Heidelberg 2010

strongly related to automated design problems because our spring systems are subject to the usual rules of real-world physics and the output of our learning algorithm can always be directly transformed into a real-world hardware spring network performing precisely the same tasks as the virtual system.

Here is an outline of the paper. In Sect. 2, the model of spring system and its dynamics are presented. We use rigid 2D graphs with suitable properties to represent spring systems. The energy function of a spring system is also discussed in Sect. 2. Formal definitions of control units/input nodes and effectors/output nodes as well as an iterative gradient-descent learning algorithm for spring systems are presented in Sect. 3. The proof of the feasibility of shape control problem is outlined in Sect. 4. Finally, we address the question how our adaptive procedure works in practice, what is the quality of results obtained and the efectiveness of the learning algorithm developed. This is discussed in details in Sect. 5.

2 The Model and Its Dynamics

Formally, we represent a spring system as an undirected graph $\mathcal{G} := (\mathcal{V}, \mathcal{E})$ with vertex/node set $\mathcal{V} \subset \mathbb{R}^d$ and edge/spring set $\mathcal{E}$. As already mentioned above, all our working examples are so far confined to the two dimensional case $d = 2$ and thus so do we restrict our theoretical discussion for presentational convenience. In this setting the graph $\mathcal{G}$ is often required to be planar so that it admits an embedding into $\mathbb{R}^2$ with its edges non-intersecting. The coordinates of a vertex $v \in \mathcal{V}$ are denoted by $(x_v^{(1)}, x_v^{(2)})$. With each edge $e = \{u, v\} \in \mathcal{E},\ u, v \in \mathcal{V}$ we associate its equilibrium (rest) length $\ell_0[e]$ and we write $\ell[e]$ for its actual length

$$\ell[e] := \mathrm{dist}(u, v) = \sqrt{(x_u^{(1)} - x_v^{(1)})^2 + (x_u^{(2)} - x_v^{(2)})^2}$$

induced by the planar embedding of the vertex set, which clearly may differ from $\ell_0[e]$. Whenever two nodes u and v are connected by an edge, we write $u \sim v$. Moreover, the spring constant $k[e] \geq 0$ is ascribed to each edge $e \in \mathcal{E}$, determining the elastic properties of the spring represented by the edge e. The energy (Hamiltonian) of a spring system configuration $\bar{x}_{\mathcal{V}} := ((x_v^{(1)}, x_v^{(2)})_{v \in \mathcal{V}})$ is given by the usual formula [2]

$$\mathcal{H}(\bar{x}_{\mathcal{V}}) := \frac{1}{2} \sum_{e \in \mathcal{E}} k[e](\ell[e] - \ell_0[e])^2 \ . \tag{1}$$

Our interest in this paper is focused on ground states of this system, defined as global energy minimisers, yet for practical reasons we also have to deal with other equilibrium states corresponding to local minima of the energy (1). The Hamiltonian (1) is isometry invariant and thus the ground state can only be unique up to isometry, which is what we shall mean by ground state uniqueness in the sequel. In general, the ground state does not have to be unique even up to isometric transformations. There are several reasons for which this can happen:

1. There may exist several planar embeddings of $\mathcal{G}$ with different collections of actual lengths, all minimising $\mathcal{H}$. This source of ground state non-uniqueness is not a serious problem though as it is only present for exceptional choices of $k[\cdot]$ and $\ell_0[\cdot]$, and can be removed upon small perturbation of these parameters.
2. There may be several energy-minimising embeddings of the graph $\mathcal{G}$ with the same actual length collections, but not arising from one another by continuum edge length preserving graph motions. This problem can also be ignored for our purposes, because such minima are separated by potential barriers impassable for our gradient-descent dynamics as discussed in the sequel.
3. Finally, there may be energy-minimising configurations arising from one another by continuous lenth preserving graph motions. This happens when the set of such motions is non-trivial, that is to say essentially larger than the set of isometric (rigid) motions. By definition this is equivalent to having the graph $\mathcal{G}$ *non-rigid*. Non-rigidity leads to essential undetermination of the ground state geometry and therefore we rule out this possibility by explicitly requiring that $\mathcal{G}$ be rigid, see [1,9] for details. Note that an easy way to keep $\mathcal{G}$ rigid is to start with a segment and keep adding subsequent graph vertices always requiring that a new vertex be connected to at least two ones already present, see Sect. 4.1 in [9].

To determine the (local) equilibrium of the spring system we let it evolve in time according to standard gradient descent dynamics

$$\frac{d}{dt}\bar{x}_{\mathcal{V}} = -\nabla\mathcal{H}(\bar{x}_{\mathcal{V}}) \ . \tag{2}$$

Clearly, this is equivalent to making each vertex $v \in \mathcal{V}$ move according to

$$\frac{d}{dt}\bar{x}_v = F_v \ , \ v \in \mathcal{V} \tag{3}$$

where

$$F_v := \sum_{u \sim v} k[\{u,v\}](\ell[\{u,v\}] - \ell_0[\{u,v\}])\frac{v-u}{\mathrm{dist}(v,u)} \tag{4}$$

is the usual elastic force acting upon a node $v \in \mathcal{V}$. In physical terms this corresponds to the evolution of our spring system in high friction environment, with the entire kinetic energy immediately dissipated. Taking into account that the dynamics (2) leads the system to equilibrium which is not always the global energy minimum, we shall write $G[\bar{x}_{\mathcal{V}}]$ to denote the unique equilibrium state reached by our gradient descent evolution initiated at $\bar{x}_{\mathcal{V}}$.

In the sequel we shall often *freeze* positions of certain collections of nodes in $\mathcal{V}$. Some of those *frozen* nodes will be interpreted as control nodes with positions set by external intervention, such as user interaction; some further nodes will be *immobilised* to reduce the number of degrees of freedom enjoyed by the system, usually in order to prevent its erratic movements during relaxation. In terms of our basic dynamics (2) this simply means taking the derivative with respect to mobile nodes only, in terms of (3) we only move vertices which are not declared frozen/immobilised.

3 System Adaptation for Shape Control

The principal shape control problem considered in this paper is put as follows. The set of spring system nodes $\mathcal{V}$ is partitioned into

1. Set $\mathcal{V}_{\text{in}}$ of control nodes. These are nodes whose position is determined by external intervention and thus is regarded as system's *input.*
2. Set $\mathcal{V}_{\text{out}}$ of observed nodes whose positions are regarded as system's *output.*
3. Set $\mathcal{V}_{\text{fixed}}$ of immobilised nodes whose positions are kept fixed in the course of system's evolution.
4. The remaining set $\mathcal{V}_*$ of auxiliary movable nodes, usually constituting the vast majority of system's vertices.

We usually require that the set of control nodes be rich enough to uniquely determine the equilibrium $G[\bar{x}_{\mathcal{V}}]$ which allows us to write $G[\bar{x}_{\mathcal{V}}] = G[\bar{x}_{\mathcal{V}_{\text{in}}}]$. In practice, this requirement is sometimes relaxed to having possible alternative equilibria given $\bar{x}_{\mathcal{V}_{\text{in}}}$ separated from the actual one $G[\bar{x}_{\mathcal{V}}]$ by high potential barriers impassable for our gradient descent dynamics. To proceed, we assume that a set $(E^{(i)})_{i=1}^{N}$ of *training examples* is given, each example $E^{(i)} := (\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}, \bar{y}^{(i)}_{\mathcal{V}_{\text{out}}})$ consisting of

1. input part $\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}$ specifying the locations of input nodes,
2. and output part $\bar{y}^{(i)}_{\mathcal{V}_{\text{out}}}$ specifying the desired locations of output nodes.

Roughly speaking, our goal is to find parameters $k[e]$ and $\ell_0[e],\ e \in \mathcal{E}$ so that the positions $\bar{x}_{\mathcal{V}_{\text{out}}}[G[\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}]]$ of output vertices in equilibrium $G[\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}]$, reached upon setting $\bar{x}_{\mathcal{V}_{\text{in}}} := \bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}$, be as close as possible to the desired locations $\bar{y}^{(i)}_{\mathcal{V}_{\text{out}}}$. To this end, we define the mean squared error function

$$\Phi = \Phi[(k[e], \ell_0[e])_{e\in\mathcal{E}}] := \frac{1}{N}\sum_{i=1}^{N} \Phi^{(i)}$$

where

$$\Phi^{(i)} := \sum_{v\in\mathcal{V}_{\text{out}}} \operatorname{dist}(y^{(i)}_v, x_v[G[\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}]])^2.$$

The adaptation of parameters $k[e]$ and $\ell_0[e]$ is performed according to the following gradient descent scheme:

1. Fix a small learning constant ρ,
2. At each step, choose (cyclically or by random) a subsequent example $E^{(i)}$ and do
   (a) Set $k[e] := -\rho\frac{\partial\Phi^{(i)}}{\partial k[e]},\ e \in \mathcal{E}$,
   (b) Set $\ell_0[e] := -\rho\frac{\partial\Phi^{(i)}}{\partial \ell_0[e]},\ e \in \mathcal{E}$.
3. Iterate 2. until Φ reaches a satisfactory value.

The gradient $\nabla\Phi^{(i)}$ can be calculated explicitly given (1) and using a second order approximation of the Hamiltonian $\mathcal{H}$ at equilibrium $G[\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}]$. However, this turns out to be quite inefficient as requiring inversion of large matrices. On the other hand, since the equilibrium $G[\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}]$ has to be found anyway, applying the dynamics (2), it is quite easy to approximate the sought gradient by directly examining the displacements of the equlibrium under small perturbations of parameters $k[e], \ell_0[e],\ e \in \mathcal{E}$. This is the option we have chosen in our implementation.

In context of the so specified task and its proposed solution, two natural questions emerge. The first one concerns the feasibility of the problem: given a set of training examples, is it always possible to find appropriate parameters $k[e], \ell_0[e],\ e \in \mathcal{E}$, so that the error Φ can be made very small ? Clearly, the answer is no if our graph is not rich enough, evidently including the extreme case $\mathcal{V}_* = \emptyset$. However, when we reformulate this question, asking whether *there exists* big enough graph so that the parameters $k[\cdot], \ell_0[\cdot]$, can be duly adjusted providing a solution for our shape control problem, the answer usually turns out positive, as argued in the next Sect. 4.

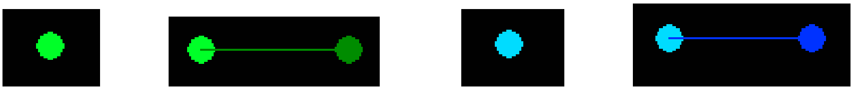
The second question to be asked is how the adaptive procedure developed in this section works in practice, what is the quality of results obtained and the effectiveness of learning algorithm. This is discussed in details in Sect. 5 below.

4 Feasibility of Shape Control Problems

In this section we briefly discuss the question of feasibility for shape control problems as put in our paper. Although it is clear that complicated tasks cannot be accomplished by a simple spring system, we claim that for most reasonable shape programming tasks a complex enough system can be found carrying them out. To see it, consider a collection $(E^{(i)})_{i=1}^N$ of *training examples*, with the notation $E^{(i)} := (\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}, \bar{y}^{(i)}_{\mathcal{V}_{\text{out}}})$ as above. Assuming in addition that vectors $\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}$ are linearly independent, which is not a restrictive condition as soon as the cardinality of $|\mathcal{V}_{\text{in}}| \geq N$, we readily see that there exists a matrix A such that $\bar{y}^{(i)}_{\mathcal{V}_{\text{out}}} = A\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}$. It remains therefore to show that a mechanical spring system performing multiplication by the matrix A can be designed. Noting that suitably adjusting the physical parameters of springs in the network we can simulate an arbitrary mechanical device, it is now enough to resort to the historical theory [7] of mechanical computing machines, which were able to perform much more than mere matrix multiplication. It should be noted though that this is an extremely complicated solution to the considered shape programming problems, only conceived for purely theoretical proof purposes and with no doubt vastly outperformed by much simpler systems, for instance those produced by our software.

5 Implementation and Example Applications

The learning algorithm for spring systems as introduced in Sect. 3 has been implemented in the programming language D [11]. Below we present figures output



Auxiliary node $v \in \mathcal{V}_*$

Node $v \in \mathcal{V}_{\text{in}}$ at current location, with desired location $\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}$

Immobilised node $v \in \mathcal{V}_{\text{fixed}}$

Node $v \in \mathcal{V}_{\text{out}}$ at current location, with desired location $\bar{y}^{(i)}_{\mathcal{V}_{\text{out}}}$

Fig. 1. Graphical symbols and colours used to visualise individual nodes of spring systems

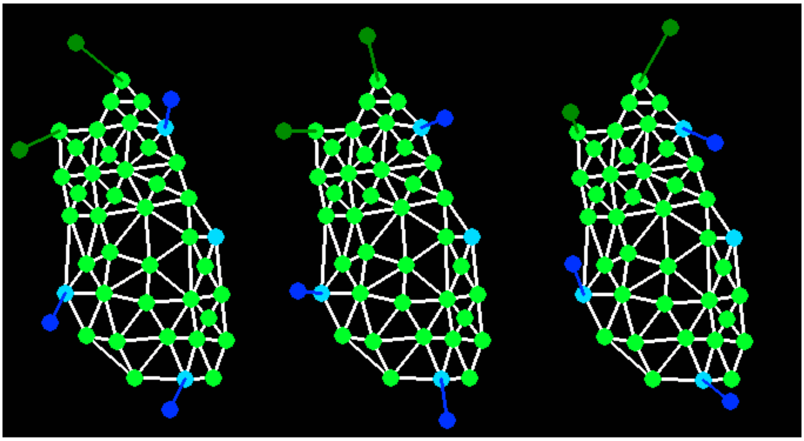


Fig. 2. Spring system before learning with training examples $(E^{(i)})_{i=1}^{3}$, respectively

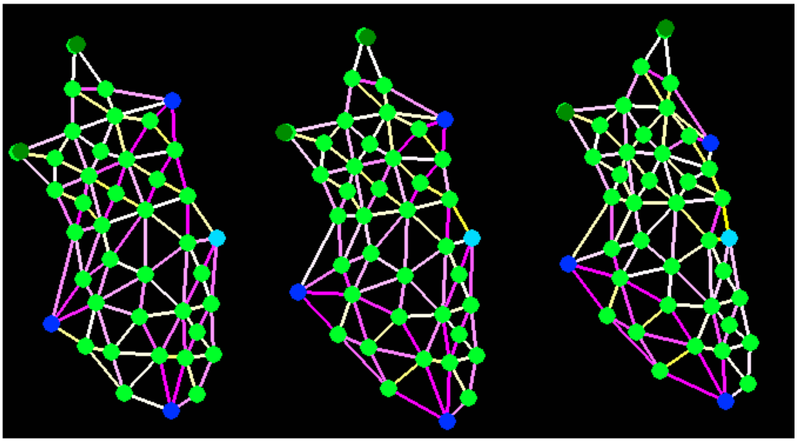


Fig. 3. Spring system after learning, in equilibria $(G[\bar{y}^{(i)}_{\mathcal{V}_{\text{in}}}])_{i=1}^{3}$ for training examples $(E^{(i)})_{i=1}^{3}$, respectively

Table 1. Spring system with its parameters

No. of spring system	$\|\mathcal{V}_{\mathrm{in}}\|$	$\|\mathcal{V}_{\mathrm{out}}\|$	$\|\mathcal{V}_{\mathrm{fixed}}\|$	$\|\mathcal{V}_{*}\|$	No. of training examples	Error Φ before learning	Average learning time to reach error $\Phi = 100$	Avarage learning time to reach error $\Phi = 25$
1	3	2	5	31	4	19557	43s	125s
2	3	2	3	66	2	3479	103s	220s
3	5	2	1	38	4	20670	106s	123s

by our software and discuss the effectiveness of the algorithm. In Fig. 1 we show the graphical symbols and colours used to visualise individual nodes of spring systems processed. Springs are represented by lines, a spring in equilibrium state has white colour, during stretching its colour is gradually becoming red, in turn during compression its colour is gradually becoming yellow.

For visualisation purposes we consider a system with three training examples $(E^{(i)})_{i=1}^{3}$ as depicted in Fig. 2 in pre-learning state. Next, in Fig. 3 we present the same system after the learning process in respective equilibria $(G[\bar{y}^{(i)}_{\mathcal{V}_{\mathrm{in}}}])_{i=1}^{3}$.

The execution times needed by our software to reach the error $\Phi = 100$ (as compared to the initial error of the order 10^4) and $\Phi = 25$ (again with the initial error of the order of 10^4) are given in Table 1 for different system sizes and learning task complexities. The computer architecture used to make all tests was AMD Athlon XP 2800+ (clock speed 2 GHz) and 512 MB RAM. Our algorithm appears to be efficient enough to solve medium-sized problems in a reasonable time (of the order of 50 nodes and 4 training examples within 2-3 minutes). Further computational optimisations are envisioned combined with use of more advanced computer architectures in order to deal with large-size realistic 3D spring systems, which is the subject of our current work in progress.

6 Conclusions

In this paper we have proposed a new gradient-descent training algorithm for shape-control tasks carried out using complex spring systems. Given a shape-programming problem, consisting of a collection of examples specifying desired input-output relationship, the algorithm outputs physical parameters (lengths, elastic constants) of a spring system designed to solve the problem. We have implemented this algorithm and tested it on medium-sized examples, concluding that it is reasonably effective there, yet requires further optimisation to cope with realistic large-sized systems in full 3D. These optimisations are the subject of our ongoing work in progress and the applications of our algorithm to complex real-world shape-programming tasks will be discussed in a separate upcoming paper.

References

1. Connelly, R., Whiteley, W.: Second-Order Rigidity and Prestress Stability for Tensegrity Frameworks. SIAM Journal of Discrete Mathematics 9, 453–491 (1996)
2. Connelly, R.: Rigidity and energy. Inventiones Mathematicae 66, 11–33 (1982)
3. Gusev, A.A.: Finite Element Mapping for Spring Network Representations of the Mechanics of Solids. Phys. Rev. Lett. 93, 034302 (2004)
4. Jagota, A., Bennison, S.J.: Spring–Network and Finite–Element Models for Elasticity and Fracture. In: Bardhan, K.K., Chakrabarti, B.K., Hansen, A. (eds.) Proceedings of a Workshop on Breakdown and Non-Linearity in Soft Condensed Matter. Lecture Notes in Physics. Springer, Heidelberg (1994); conference held at the Saha Institute for Nuclear Physics, Calcutta, India, December 1-9 (1993), pp. 186–201, vol. 437. Springer, Heidelberg (1994)
5. Kanellos, A.: Topological Self-Organisation: Using a particle-spring system simulation to generate structural space-filling lattices. Masters thesis, UCL (University College London) (2007)
6. Kellomäki, M., Aström, J., Timonen, J.: Rigidity and Dynamics of Random Spring Networks. Phys. Rev. Lett. 77, 2730 (1996)
7. Kidwell, P.A., Williams, M.R.: The Calculating Machines: Their history and development. Massachusetts Institute of Technology and Tomash Publishers, USA (1992); Translated and edited from Martin, E.: Die Rechenmaschinen und ihre Entwicklungsgeschichte. Pappenheim, Germany (1925)
8. Kilian, A., Ochsendorf, J.: Particle–Spring Systems for Structural Form Finding. Journal of the International Association for Shell and Spatial Structures: IASS 46 (2005)
9. Olfati-Saber, R., Murray, R.M.: Graph Rigidity and Distributed Formation Stabilization of Multi-Vehicle Systems. In: Proc. of the 41st IEEE Conf. on Decision and Control, Las Vegas, Nevada, (2002)
10. Ostoja–Starzewski, M.: Lattice Models in Micromechanics. Appl. Mech. Rev. 55, 35–60 (2002)
11. http://digitalmars.com/d/

Iterated Local Search for de Novo Genomic Sequencing

Bernabé Dorronsoro[1], Pascal Bouvry[1], and Enrique Alba[2]

[1] University of Luxembourg, Luxembourg
{bernabe.dorronsoro,pascal.bouvry}@uni.lu
[2] University of Málaga, Spain
eat@lcc.uma.es

Abstract. The sequencing process of a DNA chain for reading its components supposes a complex process, since only small DNA fragments can be read nowadays. Therefore, the use of optimization algorithms is required to rebuild a single chain from all the small pieces. We address here a simplified version of the problem, in which no errors in the sequencing process are allowed. The methods typically used in the literature for this problem are not satisfactory when solving realistic size instances, so there is a need for new more efficient and accurate methods. We propose a new iterated local search algorithm, highly competitive with the best algorithms in the literature, and considerably faster.

1 Introduction

The use of computational methods for the analysis of genetic material is becoming popular. The main goal of any genomic project is to determine the complete genome sequence and its genetic content. This is done in two main steps: the genome sequencing, and its annotation.

The current technology does not allow to sequence complete DNA strings. Only small pieces of up to 400 (Illumina and 454 GS FLX sequencing technologies) to around 700 (Sanger dideoxy sequencing method) nucleotids length can be read. Hence, DNA chains must be split into small portions that can be read by the current existing techniques. This process is called the *shotgun sequencing* process. Its main drawback is that the fragments orientation is lost, and thus the DNA fragment assembly problem [13], involving the process of re-assembling these small fragments for obtaining the original DNA chain, arises.

This problem has been faced by different heuristics and metaheuristics in the literature, but due to its importance and complexity, new more accurate and faster techniques are still needed. PALS [2] (Problem Aware Local Search) is a simple fast and accurate heuristic recently proposed for this problem. However, the solutions provided by PALS are not totally satisfactory when facing large problem instances. Hence, we proposed in [6] the hybridization of an efficient decentralized genetic algorithm with PALS. As a result, it was demonstrated that the combination of the two algorithms provides high quality solutions, outperforming all the other compared algorithms.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 428–436, 2010.
© Springer-Verlag Berlin Heidelberg 2010

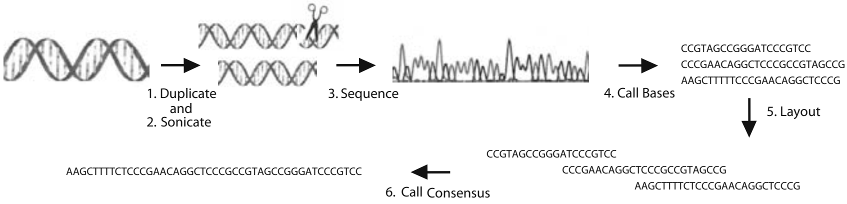

1. Duplicate
and
2. Sonicate
3. Sequence
4. Call Bases
CCGTAGCCGGGATCCCGTCC
CCCGAACAGGCTCCCGCCGTAGCCG
AAGCTTTTTCCCGAACAGGCTCCCG
5. Layout
CCGTAGCCGGGATCCCGTCC
CCCGAACAGGCTCCCGCCGTAGCCG
AAGCTTTTCTCCCGAACAGGCTCCCG
6. Call Consensus
AAGCTTTTCTCCCGAACAGGCTCCCGCCGTAGCCGGGATCCCGTCC

Fig. 1. Graphical representation of DNA sequencing and assembly

In the this work, we continue the study made in [6] on the design of new algorithms for the *de novo* genomic sequencing. Our main contribution in this paper is the new iterated local search (ILS) algorithm hybridized with PALS we propose, notably faster than the best existing algorithms in the literature, and finding solutions of better (or similar in some cases) quality.

The structure of this paper is as follows. In the next section, our problem is briefly described. Next, Section 3 presents the design aspects of the algorithm we propose. The results obtained are presented and discussed in Section 4, and we compare them versus other well known algorithms in the literature. Finally, our main conclusions are summarized in Section 5.

2 The DNA Fragment Assembly Problem

The DNA fragment assembly problem (DNA FAP) is a combinatorial optimization problem that was defined in the frame of the genome sequencing process. It starts by splitting the DNA sequence into many small sub-sequences. This is done by duplicating the original DNA sequence and cutting the different copies in random points. The resulting biological material can now be processed by computers using the current available technology. These steps are made in biological laboratories, and match with the first stages (1 to 4) shown in Fig. 1. Once the fragments are read, the assembly process is applied. It lies on computing the overlapping, the fragments order, and the consensus sequence (last two steps in Fig. 1). We are considering here an ideal case in which no errors occur in the sequencing step, commonly assumed in the specialized literature [4,14].

Since this process is one of the first steps made in any genomic project, its results are required to be highly accurate for the right functioning of the other phases of the project. This problem arises as a very complex combinatorial optimization problem (NP-Complete [12]). Its difficulty is due to the high dimensionality search space it has, which supposes to explore $2 \cdot k \cdot k!$ possible solutions in the worst case (k is the number of fragments).

For measuring the quality of a consensus sequence we use the coverage distribution. The coverage of a given location is defined as the number of fragments in that position. It is a measure of the data redundancy, and it shows the number of fragments, on average, in which a given nucleotide is expected to appear in the target DNA sequence. It can be computed as the number of bases in the fragments over the total length of the target chain [13]:

$$\text{Coverage} = \frac{\sum_{i=1}^{n} \text{ length of fragment } i}{\text{target sequence length}} \ , \tag{1}$$

where n is the number of fragments in the target sequence. The coverage is usually in the range between 6 and 10 [9]; and the higher its value, the fewer the number of gaps, and the better the obtained result.

3 The Proposed Algorithm

We present in this paper several new iterated local search (ILS) algorithms for solving the DNA FAP. In Section 3.1 we discuss the basic skeleton of the ILS algorithms we propose, while Section 3.2 describes the different algorithmic versions we considered for our tuning process.

3.1 General Description of Our ILS Algorithm

This section presents the basic skeleton of ILS we have used in this paper. Its pseudocode is shown in Algorithm 1, and it was implemented using the JCell framework, publicly available at [5].

The ILS algorithm is a simple trajectory-based metaheuristic, it starts by randomly generating and evaluating an initial solution. Then, the perturbation, improvement, and acceptance methods are iteratively applied for exploring the search space in order to improve the solution. The algorithm finishes when the termination condition is met.

The perturbation and improvement methods are essential aspects determining the good functioning of an ILS algorithm. That is why we elaborate a careful study of the behavior of the algorithm using different parameterizations for the two methods. In Section 3.2, we propose four different perturbation methods that are later experimentally compared in Section 4.2.

The improvement method we use is the application of the PALS heuristic [2], a very efficient technique for the DNA FAP. Basically, this algorithm consists of iteratively improving the solution by applying all the possible 2-Opt movements [3]. There are two important issues in PALS. One is that it does not need to perform function evaluations, since it computes the difference in the fitness value after every modification. The other one is that it is considering the number

Algorithm 1. The proposed ILS algorithm

```
sol = $CreateInitialSolution()$;
$EvaluateSolution$(sol);
bestSol = sol;
while !$StopCondition()$ do
    $PerturbationMethod$(sol);
    $EvaluateSolution$(sol);
    $PALS$(sol); // Improvement method
    if sol > bestSol then
        bestSol = sol;
    end if
end while
```

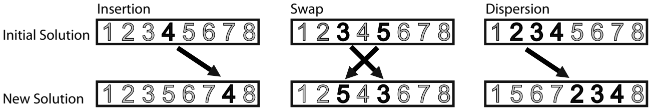


Fig. 2. Operations studied for the perturbation method

of contigs in the solution during the optimization by estimating the number of created or destroyed contigs when manipulating the tentative solutions. In contrast, most of the algorithms in the literature either do not consider the number of contigs in the solutions (what could lead to the undesirable situation in which one solution could be better than another one but having a higher number of contigs) or compute the number of contigs of the new solutions by applying an additional final refining step using a greedy heuristic [10].

Finally, the acceptance method used in our ILS simply consists in always accepting all the new generated solutions. We adopted this strategy after observing in several experiments a fast convergence of solutions to local optima (typically, after one or two iterations). Hence, since PALS is deterministic, it will not be able to improve the solution once it has converged to a local optimum.

3.2 Studied ILS Designs

We are testing in this work four different configurations for the perturbation method. They use different combinations of the *insertion*, *swap*, and *dispersion* operators. These operators are commonly used in the literature for introducing diversity in routing, which have some similarities with the DNA FAP [11].

In Fig. 2 we show the three studied operators. The *swap* operator is to randomly select two fragments in our solution and exchange their positions. The *insertion* operator lies in randomly choosing a single fragment and inserting it in a new random position, displacing all the fragments between the old and new positions. Finally, the *dispersion* operator is a generalization of *insertion* in which not only one fragment is moved to a new location but a chain of consecutive fragments. The length of this chain is random, but it will never be larger than the distance between the first fragment and the destination positions, since in other case no change would be made after applying the method.

We now proceed to present the ILS algorithms we designed with the different perturbation methods. We took the decision of including the insertion operator in all the perturbation methods due to our previous experience with this kind of problems [1]. The four ILS algorithms studied are listed below:

- ILS-I: It uses only the insertion operation in the perturbation method.
- ILS-IS: Insertion operation is applied 50% of the cases, as well as swap.
- ILS-ID: Insertion is applied 50% of the cases, as well as displacement.
- ILS-IDS: The three operators are applied with 33.3% probability.

The perturbation method is applied with probability $p_p = 1/$ No. of Fragments. When the method is composed by more than one operator, they will be applied with equal probability. This means that one solution can be modified by different operators at the same time.

4 Experimentation

We present in this section the results obtained with the ILS algorithms. They were implemented in Java (using JCell [5] framework), and they were run under Linux on an Intel Xeon 2.0GHz processor (using only one core). The problems studied are described in Section 4.1. Then, we analyze the influence of the different proposed parameterizations in Section 4.2, and compare them to those of an state-of-the-art algorithm for the considered problem.

4.1 Problem Instances

We evaluate the algorithms on the two largest instances of the problem studied in [2]. They correspond to the *Neurosphora crassa* bacterial artificial chromosome (BAC), a kind of fungus typically found in common loaf composed by 77,292 bases. It was taken from the NCBI[1] website, and the fragments were generated with GenFrag [7]. We study two instances: BX842596(4), composed of 442 fragments of 708 basis each, having coverage 4; and BX842596(7), having 773 fragments of length 703, and coverage 7. These instances are lager and more complex than those usually tackled by other researchers (which are usually around 15 or 30 thousands of bases length, or even less [8]).

The two studied instances are really hard to solve, since they were generated from very long sequences using an small/medium coverage value, and a very restrictive cutting (threshold for joining adjacent fragments into the same contig), having value 30.

4.2 Parameter Tuning Process

We proceed in this section to compare and evaluate the different algorithmic versions proposed. The results are shown in tables 1 and 2 for the ILS with the four different perturbation methods (described in Section 3.2), and three different intensities of PALS for each of them (5, 50 and 500 iterations). In total, we are comparing 12 different versions of our ILS algorithm for each of the two considered problems. The values we report in the tables are the average fitness value of solutions (which is a measure of the overlapping degree among fragments composing the solution), the average number of contigs in these solutions, and the average time needed by the algorithm (in seconds), all these values are presented together with their standard deviations. They were obtained after 100 independent runs for each algorithm and instance. In total, we made 2400 experiments. The overall best values in each table are marked with a grey color background.

In order to obtain concluding results, we have performed statistical tests on the results. For that, we first check whether the data follow a normal distribution

[1] http://www.ncbi.nlm.nih.gov

Table 1. Comparison of the different ILS algorithms proposed. Instance BX842596(4).

Perturb.	5 steps of PALS			50 steps of PALS			500 steps of PALS			Test
	Solution	Contigs	Time	Solution	Contigs	Time	Solution	Contigs	Time	
ILS-I	227663.42 ±92.61	1.05 ±0.30	70.39 ±1.89	227638.08 ±104.69	1.00 ±0.00	81.29 ±2.23	227644.86 ±109.47	1.00 ±0.00	81.11 ±2.41	−,−,+
ILS-IS	227557.94 ±142.84	1.02 ±0.12	61.87 ±1.61	227581.80 ±140.20	1.00 ±0.00	68.55 ±2.47	227614.23 ±131.29	1.00 ±0.00	68.93 ±2.10	+,−,+
ILS-ID	227668.62 ±93.81	1.01 ±0.10	69.22 ±2.43	227659.59 ±94.62	1.00 ±0.00	78.52 ±2.72	227640.29 ±104.08	1.00 ±0.00	79.43 ±2.22	−,−,+
ILS-IDS	227588.61 ±134.35	1.04 ±0.24	64.29 ±1.58	227639.61 ±123.68	1.00 ±0.00	72.08 ±2.02	227624.17 ±119.97	1.01 ±0.10	72.06 ±1.93	+,−,+
Test	+	−	+	+	−	+	−	−	+	

Table 2. Comparison of the different ILS algorithms proposed. Instance BX842596(7).

Perturb.	5 steps of PALS			50 steps of PALS			500 steps of PALS			Test
	Solution	Contigs	Time	Solution	Contigs	Time	Solution	Contigs	Time	
ILS-I	444995.94 ±174.72	1.03 ±0.17	249.85 ±9.90	445076.36 ±151.53	1.00 ±0.00	292.36 ±9.28	445041.22 ±213.98	1.00 ±0.00	309.93 ±13.06	+,−,+
ILS-IS	444774.15 ±341.76	1.00 ±0.00	217.01 ±6.92	444860.78 ±303.20	1.00 ±0.00	257.32 ±10.20	444868.18 ±290.16	1.00 ±0.00	259.44 ±10.80	−,−,+
ILS-ID	445020.42 ±225.67	1.00 ±0.00	242.46 ±7.57	445058.02 ±176.78	1.00 ±0.00	297.96 ±13.30	445059.88 ±194.41	1.00 ±0.00	297.30 ±12.96	−,−,+
ILS-IDS	444946.95 ±291.06	1.00 ±0.00	233.05 ±8.66	444999.36 ±198.48	1.00 ±0.00	268.77 ±11.0	444984.65 ±190.63	1.00 ±0.00	272.54 ±12.24	−,−,+
Test	+	−	+	+	−	+	+	−	+	

or not using the Shapiro-Wilks test. Then, if the data are normally distributed we perform an ANOVA test. In other case, the tool we use is the Kruskal-Wallis test. This statistical study allows us to asses if there are meaningful differences among the compared algorithms with 95% probability. In the last row in tables 1 and 2 we show whether there are statistically significant differences (‘+’, when p-value ≤ 0.05) or not (‘−’, when p-value > 0.05) among the values in the corresponding column. Additionally, in the right hand column we show the differences we obtained after applying 5, 50, and 500 steps of PALS in terms of the solutions found, the number of contigs, and the time, respectively.

In Table 1 we show the results obtained for instance BX842596(4). The first interesting result we see is that there are not statistically significant differences in terms of the number of contigs in any case. The algorithms usually find solutions of only one contig (this is precisely the goal), and only a maximum of 2% of the executions with 5 steps of PALS cannot find a single contig solution.

If we now pay attention to the effects of incrementing the number of steps of PALS, we can generally see that there are not statistically significant differences in terms of the solution found. The exception are the methods in which the swap operation is involved (ILS-IS and ILS-IDS), since they are statistically worse than the others in some cases. Regarding the execution time, those algorithms implementing 5 steps of PALS are significatively better than the others. Additionally, the fact that there are not important differences between the algorithms using 50 and 500 steps of PALS caught our attention. This tells us that 50 steps of PALS are usually enough in our algorithm for converging to a local optimum in the improvement method. Finally, we emphasize that all the algorithms found the same best fitness value, in at least one of the 100 runs.

Table 3. Comparison among ILS-D and SACMA

		Solution		No. of Contigs		Time
		Best	**Average**	**Best**	**Average**	**(Seconds)**
BX842596(4)	SACMA	227779	227768, 69 ± 31.09	1	1, 00 ±0,00	154.85 ±14,55
	ILS-D	227779	227638, 08 ± 104,69	1	1, 00 ±0,00	81.29 ± 2,23
	Test		+		+	+
BX842596(7)	SACMA	445411	445324, 97 ± 68.81	1	1, 01 ±0,10	651.20 ±85.11
	ILS-D	445339	445076, 36 ± 151,53	1	1, 00 ±0,00	292, 36 ± 9,28
	Test		+		+	+

Regarding the perturbation method, we can say that ILS-ID is the algorithm obtaining the best quality solutions, since it is the best algorithm when using 5 and 50 steps of PALS, and the second best one in the case of 500 PALS steps (with no statistical difference with the first one). In contrast, this is the second worst algorithm in terms of run time, followed by ILS-I. The differences in terms of time are not high (all the algorithms are within an interval of about 10 seconds for each PALS configuration), but they are statistically significant in all the cases. In this case, the algorithms implementing the swap operation are faster, but they are the worst ones in terms of the solution quality.

As in the case of the small instance, the differences between all the algorithms for instance BX842596(7) are not significant in terms of the number of contigs. Indeed, as it can be seen in Table 2, in this case all the algorithms find in every run a single contig solution, with the exception of ILS-I with 5 steps of PALS, that could not find it in 2% of the runs. In terms of time we see similitudes with the results obtained for the small instance too. The time is significantly shorter in the case of using 5 steps of PALS than in the equivalent algorithms with 50 and 500 steps, although there are not important differences between these two last versions (with the only exception that ILS-I with 50 steps of PALS is faster than the version using 500 steps). For this problem there are not significant differences among the algorithms with the three different versions of PALS. The only exception is ILS-I, statistically worse with 5 steps than with 50 or 500.

Regarding the perturbation method used, the only significant differences we obtained in terms of the solutions quality are given by the methods using the swap operator, which are worse than the others, in general. As it happened for the small instance, the time differences are always significant, with the exception of ILS-I and ILS-ID, employing similar times for solving the problem.

Summarizing, it stands out that most of the algorithms found single contig solutions. This is important because it is one of the main goals for solving the problem. Regarding the overlapping degree among fragments (fitness value), the algorithms ILS-I and ILS-ID (those not implementing swap operation) are the best ones, in general. However, they require longer run times, although the difference with the fastest compared algorithm is about 10% slower. Hence, in order to obtain better quality solutions, slightly slower algorithms are required. Since the solutions quality is a key aspect in this problem, we choose ILS-I and ILS-ID as the best compared algorithms. There are not statistical significant differences between these two algorithms in any of the considered parameters (overlapping degree of fragments, number of contigs, and time).

To end this section, we compare our results with SACMA, a state-of-the-art advanced genetic algorithm [6] hybridized with PALS. We choose ILS-I for the comparison (with 50 steps for PALS). The results are shown in Table 3. The algorithms are compared in terms of the fitness value, the number of contigs, and the time (in seconds). In order to make a fair comparison, we show the results after 100 runs and we made the same statistical study as in Section 4.2.

The results in Table 3 show significant differences in all cases, with the exception of the number of contigs in the solutions found by ILS-I and SACMA for the two problems. We can see that even when SACMA is significantly better than ILS-I in terms of the solution fitness values, the difference between the two algorithms is less than 0.06% for both instances. Moreover, the best solutions found by the two algorithms has the same value for the small instance and a difference of 0.016% for the big one.

Regarding the number of contigs in the reported solutionds, both SACMA and ILS-I find similar results, finding a single contig solution in every run, with the exception of SACMA, which finds a two contigs solution in one of the runs for BX843596(7) instance.

The largest differences we found between SACMA and ILS-I are in terms of time. In this case, ILS-I is about twice faster than SACMA (the difference is 47.5% for BX843596(4) and 55.1% for BX843596(7)). Hence, ILS-I allows us to find similar results to those of SACMA but in half time.

5 Conclusions

We proposed in this paper a new iterated local search algorithm, called ILS-I, for the fragment assembly problem. The algorithm was designed using the PALS heuristic as the improvement method, and a tuning process was made studying different perturbation and improvement methods. The resulting algorithm was compared versus SACMA, a highly competitive structured population genetic algorithm hybridized with PALS too. As a result, ILS-I obtains similar results than those found by SACMA, in half time.

Among the main next research lines emerging from this work, we plan to study larger instance sizes than the ones studied here. We think that some of the differences shown in this paper could be more important in the case of solving more difficult and larger problem instances. Moreover, we plan to extend our study to a more realistic case of the problem in which we would need to deal with errors in the reading process. This way, we would be solving instances with much more scientific interest.

References

1. Alba, E., Dorronsoro, B.: Cellular Genetic Algorithms. In: Operations Research/Compuer Science Interfaces. Springer, Heidelberg (2008)
2. Alba, E., Luque, G.: A New Local Search Algorithm for the DNA Fragment Assembly Problem. In: Cotta, C., van Hemert, J. (eds.) EvoCOP 2007. LNCS, vol. 4446, pp. 1–12. Springer, Heidelberg (2007)

3. Croes, G.: A method for solving traveling salesman problems. Op. Res. 6, 791–812 (1958)
4. Dohm, J.C., Lottaz, C., et al.: SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing. Gen. Res. 17, 1697–1706 (2007)
5. Dorronsoro, B.: The jcell framework (2007), `http://neo.lcc.uma.es/Software/JCell/`
6. Dorronsoro, B., Alba, E., Luque, G., Bouvry, P.: A Self-Adaptive Cellular Memetic Algorithm for the DNA Fragment Assembly Problem. In: Proc. of the IEEE Congress on Evolutionary Computation (CEC), pp. 2656–2663 (2008)
7. Engle, M.L., Burks, C.: GenFrag 2.1: New Features for More Robust Fragment Assembly Benchmarks. Bioinformatics 10(5), 567–568 (1993)
8. Kikuchi, S., Chakraborty, G.: Heuristically tuned GA to solve genome fragment assembly problem. In: IEEE CEC 2006, pp. 1491–1498 (2006)
9. Kim, S.: A structured Pattern Matching Approach to Shotgun Sequence Assembly. PhD thesis, Computer Science Department, The University of Iowa (1997)
10. Li, L., Khuri, S.: A Comparison of DNA Fragment Assembly Algorithms. In: Int. Conf. METMBS, pp. 329–335 (2004)
11. Parsons, R., Forrest, S., Burks, C.: Genetic algorithms, operators and DNA fragment assembly. Machine Learning 21, 11–33 (1995)
12. Pop, M.: Shotgun sequence assembly. Advances in Computers 60, 194–248 (2004)
13. Setubal, J., Meidanis, J.: Introduction to Computational Molecular Biology. Fragment Assembly of DNA, ch. 4, pp. 105–139. University of Campinas, Brazil (1997)
14. Zerbino, D.R., Birney, E.: Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. Genome Research 18, 821–828 (2008)

Tournament Searching Method to Feature Selection Problem

Grzegorz Dudek

Department of Electrical Engineering, Czestochowa University of Technology,
Al. Armii Krajowej 17, 42-200 Czestochowa, Poland
dudek@el.pcz.czest.pl

Abstract. A new search method to the feature selection problem – the tournament searching – is proposed and compared with other popular feature selection methods. The tournament feature selection method is a simple stochastic searching method with only one parameter controlling the global-local searching properties of the algorithm. It is less complicated and easier to use than other stochastic methods, e.g. the simulated annealing or genetic algorithm. The algorithm was tested on several tasks of the feature selection in the supervised learning. For comparison the simulated annealing, genetic algorithm, random search and two deterministic methods were tested as well. The experiments showed the best results for the tournament feature selection method in relation to other tested methods.

Keywords: feature selection, tournament feature selection, tournament searching, stochastic combinatorial optimization.

1 Introduction

The feature selection (FS) is an essential problem in the modelling of objects, processes and phenomenon, explored especially in pattern recognition, machine learning, data mining and artificial intelligence. The aim of FS is to reduce the dimension of the input vector by the feature (variable) subset selection which describes object in the best manner and ensures the best quality of the learning model. In this process the irrelevant, redundant and unpredictive features are omitted. For example, only the features, which contain the most information about the membership of the objects to the classes and which allow to minimize the risk of misclassification, are selected in pattern recognition. The other way of reduction in dimension of modelling problem relies on an application of the linear or nonlinear transformation, mapping the space of original features on the space with smaller dimension (the principal component analysis is the popular example). A creation of the new features on the base of the original features is called the feature extraction.

FS is applied due to several reasons [1], [2], [3]:

1. Model simplification. Consideration of the large number of features increases complexity of the model, what does not go with improvement in its quality. Some features can be redundant (e.g. the correlated features) or irrelevant (features not

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 437–444, 2010.
© Springer-Verlag Berlin Heidelberg 2010

carrying information about model output). The elimination of these features simplifies the model, improves its comprehensibility, stays in contrary to curse of dimensionality and clarifies the relationships between features.

2. Improvement in model quality. A removal of the unpredictive features does not often worsen the model quality, but improves it. For example a removal of features with random values, interferenced by huge noise, usually allows to improve the model operation.
3. Improvement in generalization. The simpler models with smaller number of parameters have greater generalization abilities (elimination of the noisy features reduces the model overfitting).
4. Reduction of computation time. The model created with the use of lower number of the features learns and works faster.
5. Saving of computer memory is proportional to the number of eliminated features.
6. Reduction in cost of data collection, communication, maintenance and technical realization of the model.
7. Data visualization when the number of selected features is low (1, 2 or 3).

The methods of the feature selection can be generally divided into filter and wrapper ones [4]. Filter methods do not require application of learning model to select relevant features. They select features as a preprocessing step, independent on the choice of the predictor. They also use information included in the dataset, e.g. the correlation between variables and discriminatory abilities of the individual features, to create the most promising feature subset before commencement of learning. The main disadvantage of the filter approach is the fact that it totally ignores the effect of the selected feature subset on the performance of the learning model [4]. Many approaches to the feature relevance measurements are presented in [5].

The wrapper approach operates in the context of the learning model – it uses feature selection algorithm as a wrapper around the learning algorithm and has usually better predictive accuracy than the filter approach. The wrapper approach using the learning model as a black box is remarkably universal. However, this approach can be very slow because the learning algorithm is called repeatedly. The comparative experiments between the wrapper and filter models confirmed that it is inappropriate to evaluate the usefulness of an input variable without taking into consideration the algorithms that built the classification or regression model [4]. The filter approach can be used as a preprocessing step for the wrapper approach in the analysis of huge datasets.

Some learning models have internal build-in mechanisms of the FS. For example decision trees (CART, ID3, C4.5) which incorporate FS routine as a subroutine and heuristically search the space of feature subsets along tree structure during the learning process. The approaches in which FS is an essential part of the training process are so-called the embedded methods.

The described above FS methods attempt to select the most relevant feature subset. Another approach is to apply a weighting function to features and in effect to attribute degrees of relevance to them. The well-known feature weighting method is the perceptron updating rule, which adds or subtracts weights on a linear threshold unit in response to training errors.

Many approaches were used to the FS problem: branch and bound algorithm [6], simulated annealing [7], genetic algorithms [7], [8], rough sets [9], adaptive agents [10], tabu search method [11], support vector machines [12] and ensembles [13]. The popular suboptimal fast strategies are the sequential forward selection and sequential backward selection algorithms [14], [3] based on simple greedy deterministic heuristics. The expansion of these strategies is plus l-take away r method [15] and floating search method [16]. The review of the FS subject matter can be found in [1], [2], [4], [17] and [18].

A goal of this work is to propose a simple, easy to use and control wrapper method of FS.

2 Tournament Feature Selection Method

In the FS problem the decision variables are binary and denote whether the feature is selected (1) or not (0). The solution is a binary vector composed of bits representing all m features: $\mathbf{x} = [x_1, x_2, \ldots, x_m]$.

The tournament method to feature selection (TFS) consists of exploring the solution space starting from a randomly selected solution and generating the new ones by perturbing it. When the set of new l candidate solutions $\Omega = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l\}$ is generated (l is called the tournament size), their costs (e.g. classification errors) are calculated using learning model. The best candidate solution (the tournament winner), with the lowest value of the cost function $C(\mathbf{x})$, is selected and it replaces the parent solution, even in case it is worse than the parent solution.

TFS searches the solution space by exploring neighborhood of the current solution by means of the move operator. The neighborhood is defined as $\Phi = \{\mathbf{x} \mid d_H(\mathbf{x}, \mathbf{x}^*) = 1\}$, where $\mathbf{x}^*$ is the parent solution and d_H is the Hamming distance. The move operator produces a candidate solution by switching the value of the randomly chosen bit (different for each candidate solution) of the parent solution.

If the tournament size is equal to 1, this procedure comes down to the random search method. On the other hand, when $l = m$ this method becomes the hill climbing method. $l = 1, 2, \ldots, m$ is the only parameter here. Its value decides about the exploration/exploitation properties of the algorithm, and can be fixed or can increase with the iteration number.

The TFS algorithm can be summarized in the following steps:

1. Generation of the initial solution (first parent solution) by random.
2. Generation of the set of l candidate solutions from the parent solution using the move operator.
3. Evaluation of the candidate solutions using learning model.
4. Selection of the best solution among the candidate solutions.
5. Replacement of the parent solution by the tournament winner.
6. Repeat steps 2-6 until the stop criterion is reached.

The flowchart of the TFS is shown in Fig. 1, where the flowcharts of popular stochastic FS methods, the simulated annealing (SA) and genetic algorithm (GA), are shown as well. The GA and TFS have parallel structure, the solution space is searched by a population of points. In SA solutions are generated one by one.

The evaluation procedures and move or mutation operators are similar in these algorithms, but the selection procedures are different. In the SA method the new solution **x** is either accepted or rejected according to an acceptance probability based on the Boltzmann distribution:

$$p(\mathbf{x}) = \min\left[1, \exp\left(\frac{-\Delta C}{T}\right)\right], \tag{1}$$

where T is a global time-varying parameter called the temperature, $\Delta C = C(\mathbf{x}) - C(\mathbf{x}^*)$.

The cooling schedule specifies an initial value of the temperature T_0, a temperature update function $T_{k+1} = f(T_k)$ and the inner-loop and outer-loop criterions (not shown in Fig. 1). The temperature parameter T controls the probability of acceptance. Initially the high value of temperature is selected. Then it decreases slowly according to search advance in order to provoke the convergence of the algorithm to the global optimum. In the pioneering work [19] the temperature was reduced according to the geometric cooling schedule:

$$T_{k+1} = \alpha T_k\,, \tag{2}$$

where 0<α <1 is a constant factor.

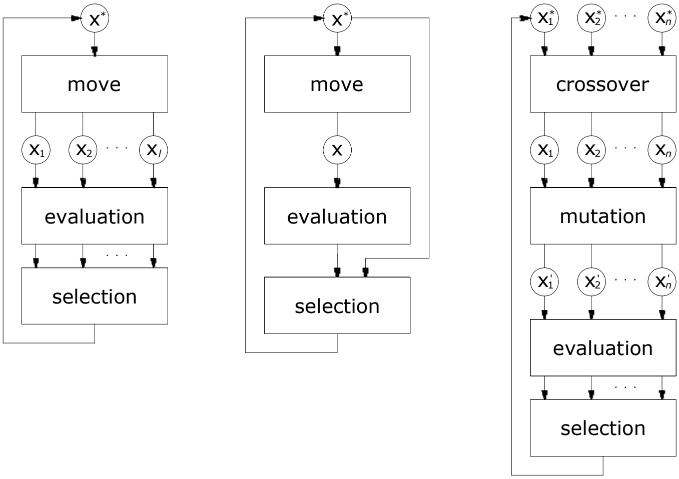


Fig. 1. Simple flowcharts of the tournament feature selection method (left), simulated annealing (middle) and genetic algorithm (right)

The GA is the more complicated method than TFS and SA. It requires many parameters to be set, i.e.: population size, probability of crossover and mutation, selection procedure parameters.

There are many methods of selection in GA, e.g.: roulette wheel selection, tournament selection, rank selection and some others. The main idea of these methods is to select the relatively good solutions from the population – the better the solution is, the more chances to be selected it has. The tournament selection procedure is similar to that used in TFS, but instead of one, many tournaments of the size l are performed. The winners form the new population. Parameter l controls the selective pressure.

All these FS methods have some built-in mechanism of escaping from the local minima/maxima. In TFS method this mechanism is very simple – the parent solution is replaced by the tournament winner, even if it represents a worse solution.

3 Experimental Comparison of Search Algorithms

The TFS method was verified on several test problems of data classification. Benchmark datasets, described in Table 1, were taken from the UCI repository [20]. The features in datasets were standardized.

Table 1. Description of data used in experiments

Dataset	Size	Features	Classes	Optimal *k* value
Ionosphere	351	34	2	3
Cancer	569	30	2	4
Heart	270	13	2	7
Wine	178	13	3	4
Glass	214	9	6	5
Diabetes	768	8	2	14

where: Cancer – the Wisconsin diagnostic breast cancer dataset; Heart – the Statlog heart dataset.

K-nearest neighbor method (k-NN) was used as a classifier, with k determined a priori for all features (optimal k values are shown in Table 1). The classification accuracy was determined in the leave-one-out procedure. For comparison, sequential forward selection (SFS) and sequential backward selection (SBS) algorithms [3], [14] were used to FS as well as the genetic algorithm, simulated annealing and random search method. For each dataset the feature space was optimized running algorithms 30-times and starting from the same initial solution (except for GA which operates on a population of solutions). The number of solutions generated in the searching process was the same for all algorithms and equal $40 \cdot \text{round}(m/2)^2$. The set of tested algorithms and the values of their parameters, which were adjusted experimentally, are listed below.

- TFS: $l = \text{round}(m/3)$.
- SA schedule proposed by Kirkpatrick et al. in [19]. The initial temperature was calculated iteratively from (1), where $p_0 = 0.99$ and ΔC was the average of the absolute value of the cost variation obtained for an initial sequence of random transitions. The value of temperature reduction coefficient α was determined in such a way that the acceptance probability of the trial solution, which is 0.01% worse from the parent solution, is equal 0.1, when the annealing process is advanced in 80%:

$$\alpha = \left(\frac{0.0043}{T_0} \right)^{\frac{1}{0.8L_{out}-1}}, \tag{3}$$

where L_{out} is the number of the outer-loop iterations, $L_{out} = 20\cdot\text{round}(m/2)$. The number of inner-loop iterations $L_{inn} = 2\cdot\text{round}(m/2)$.

- GA: selection method – binary tournament, crossover method – uniform crossover with 0.5 probability of swapping, probability of crossover – 0.9, mutation method – classical, expected number of the chromosome mutation – 1, number of generations – 20·round(m/2), population size – 2·round(m/2), elite strategy – copying the best chromosome of population t-1 into the population t.
- RS – random search algorithm in which the new solution was generated by changing one randomly chosen bit of the old solution.

The results – accuracies of the classifier using selected features (minimal, maximal and mean in 30 runs) and their standard deviations – are presented in Tab. 2. The ranking of FS methods is shown in Fig. 2, where horizontal axis represents the mean difference between the mean accuracies of the best FS method and the given FS method.

In general the effectiveness of the search process depends on the algorithm parameter values. Usually the faster convergence leads to the freezing of the searching process in the local optimum. In most cases the tuning of the algorithm is a hard job requiring many experiments. Here, the parameter values of the algorithms were set in such a way in order to ensure enough time to the broad exploration of the feature space.

Unexpectedly in all cases TFS method gave the best results. The TFS method turned out to be more resistant to the local minima traps than others stochastic search algorithms tested in this work. The main advantage of TFS is only one parameter to adjust – the tournament size l.

The second best method was the genetic algorithm – the most complicated among tested methods. All stochastic methods except for random search gave better results on average than the deterministic approaches SFS and SBS.

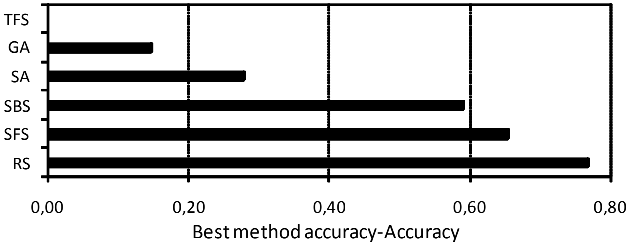


Fig. 2. Ranking of FS methods

Table 2. Percentage accuracies of the classifier: mean Acc_{mean}, minimal Acc_{min}, maximal Acc_{max}, and their standard deviations σ_{Acc}

Dataset		TFS	SA	GA	RS	SFS	SBS	Without FS
Ionosphere	Acc_{mean}	94.78	94.00	94.40	92.17	94.02	94.30	84.33
	Acc_{min}	94.59	92.88	93.73	91.45			
	Acc_{max}	95.44	94.59	95.44	93.16			
	σ_{Acc}	0.26	0.32	0.26	0.47			
Cancer	Acc_{mean}	98.25	97.97	98.11	97.57	97.72	97.54	96.61
	Acc_{min}	97.89	97.54	97.72	97.36			
	Acc_{max}	98.42	98.42	98.42	97.89			
	σ_{Acc}	0.18	0.19	0.22	0.12			
Heart	Acc_{mean}	86.25	85.92	86.07	85.67	85.93	85.93	82.22
	Acc_{min}	85.93	85.19	85.56	84.44			
	Acc_{max}	86.30	86.30	86.30	86.30			
	σ_{Acc}	0.13	0.29	0.21	0.43			
Wine	Acc_{mean}	98.88	98.62	98.69	98.35	98.88	97.75	96.07
	Acc_{min}	98.88	97.75	98.31	97.75			
	Acc_{max}	98.88	98.88	98.88	98.88			
	σ_{Acc}	0.00	0.29	0.27	0.25			
Glass	Acc_{mean}	74.77	74.75	74.77	74.67	73.36	74.77	65.89
	Acc_{min}	74.77	73.36	74.77	73.36			
	Acc_{max}	74.77	74.77	74.77	74.77			
	σ_{Acc}	0.00	0.14	0.00	0.36			
Diabetes	Acc_{mean}	77.21	77.20	77.21	77.09	76.30	76.30	73.96
	Acc_{min}	77.21	76.56	77.21	76.69			
	Acc_{max}	77.21	77.21	77.21	77.21			
	σ_{Acc}	0.00	0.08	0.00	0.19			

4 Conclusions

A practitioner would like to know which algorithm is the best choice to solve a given problem. Usually it is very hard to select one method looking on “no free lunch” theorem. There are no problem-independent reasons to favor one optimization method over another. The simulated annealing, genetic algorithm and proposed tournament searching method belong to the stochastic generic search algorithms. Their application to the optimization problem requires preliminary tests in order to tune the parameters. This is a time-consuming procedure depending on the numbers of parameters. There is only one parameter in the tournament searching, controlling the global-local search properties, which makes this algorithm easy to use.

The empirical comparison between all of the presented algorithms showed that the tournament searching method produced the best results (the accuracies of the k-NN classifier) for all datasets.

The tournament searching method, similarly to the genetic algorithm, has a parallel structure – several candidate solutions can be generated and evaluated at the same time. This results in the runtime decreasing.

In order to evaluate the tournament searching method more tests on different combinatorial optimization problems are necessary. This will be the subject of the future work.

References

1. Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. Journal of Machine Learning Research 3, 1157–1182 (2003)
2. Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L.A. (eds.): Feature Extraction. Foundations and Application. Springer, Berlin (2006)
3. Theodoridis, S., Koutroumbas, K.: Pattern Recognition. Elsevier Academic Press, Amsterdam (2003)
4. Kohavi, R., John, G.H.: Wrappers for Feature Subset Selection. Artificial Intelligence 1-2, 273–324 (1997)
5. Duch, W.: Filter Methods. In: [2]
6. Somol, P., Pudil, P., Kittler, J.: Fast Branch and Bound Algorithm in Feature Selection. IEEE Transaction on Pattern Analysis and Machine Intelligence 26, 900–912 (2004)
7. Siedlecki, W., Sklansky, J.: On Automatic Feature Selection. International Journal on Pattern Recognition and Artificial Intelligence 2(2), 197–220 (1988)
8. Raymer, M.L., Punch, W.F., Goodman, E.D., Kuhn, L.A., Jain, A.K.: Dimensionality Reduction using Genetic Algorithms. IEEE Trans. Evol. Comput. 4(2), 164–171 (2000)
9. Modrzejewski, M.: Feature Selection using Gough Sets Theory. In: Brazdil, P.B. (ed.) ECML 1993. LNCS, vol. 667, pp. 213–226. Springer, Heidelberg (1993)
10. Menczer, F., Degeratu, M., Street, W.N.: Efficient and Scalable Pareto Optimization by Evolutionary Local Selection Algorithms. Evolutionary Computation 8(2), 223–247 (2000)
11. Oduntan, I.O., Toulouse, M., Baumgartner, R., Bowman, C., Somorjai, R., Crainic, T.G.: A Multilevel Tabu Search Algorithm for the Feature Selection Problem in Biomedical Data. Computers and Mathematics with Applications 55, 1019–1033 (2008)
12. Bi, J., Bennett, K.P., Embrechts, M., Breneman, C.M., Song, M.: Dimensionality Reduction via Sparse Support Vector Machines. Journal of Machine Learning Research 3, 1229–1243 (2003)
13. Tuv, E., Borisov, A., Runger, G., Torkkola, K.: Feature Selection with Ensembles, Artificial Variables and Redundancy Elimination. Journal of Machine Learning Research 10, 1341–1366 (2009)
14. Devijver, P.A., Kittler, J.: Pattern Recognition: A Statistical Approach. Prentice-Hall, London (1982)
15. Stearns, S.: On Selecting Features for Pattern Classifiers. In: Proc. of 3rd International Joint Conf. on Pattern Recognition, pp. 71–75 (1976)
16. Pudil, P., Novovicova, J., Kittler, J.: Floating Search Methods in Feature Selection. Pattern Recognition Letters 15, 1119–1125 (1994)
17. Blum, A.L., Langley, P.: Selection of Relevant Features and Examples in Machine Learning. Artificial Intelligence 97, 245–271 (1997)
18. Kim, Y.: Feature Selection in Supervised and Unsupervised Learning via Evolutionary Search. Ph.D. Dissertation, University of Iowa (2001)
19. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by Simulated Annealing. Science 220, 671–677 (1983)
20. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html

New Linguistic Hedges in Construction of Interval Type-2 FLS*

Piotr Dziwiński[2], Janusz T. Starczewski[1,2], and Łukasz Bartczuk[2]

[1] Academy of Management (SWSPiZ), Institute of Information Technology,
ul. Sienkiewicza 9, 90-113 Łódź, Poland
janusz.starczewski@gmail.com

[2] Department of Computer Engineering, Czestochowa University of Technology,
Al. Armii Krajowej 36, 42-200 Czestochowa, Poland
{piotr.dziwinski,janusz.starczewski,lukasz.bartczuk}@kik.pcz.pl

Abstract. The paper presents a methodology for application of an interval type-2 codebook to computing with words. The crucial problem in this task is to formulate new hedge operators for fuzzy sets. The proposed hybrid system is demonstrated in a numerical example.

1 Introduction

According to Zadeh [1], "by a linguistic variable we mean a variable whose values are words or sentences in a natural or artificial language". These variables take their values from a term-set, i.e. the collection of all linguistic values. The term set is composed of so called primary terms e.g., *high* and *low*, and optionally linguistic modifiers e.g., *extremely*, *very* and *more or less*. In the Zadeh's formulation, semantic rules associating linguistic values with their meanings are expressed by fuzzy sets (FSs) and the linguistic modifiers are performed by hedge operations.

However, human descriptions of semantic rules are usually ill-defined, since "words can mean different things to different people", as Mendel pointed out in [2]. Also the linguistic hedges are very context dependent. Accordingly, linguistic values frequently are associated with the concept of fuzzy truth-values specified by linguistic values such as *true*, *rather true* or *more or less true*.

Many papers [3,4,5,6,7,8,9,10,11,12,13,14,15] adopt fuzzy logic systems (FLS) to computing with words (CWW) — a new methodology performing computations in the natural language proposed by Zadeh [16].

2 New Linguistic Hedges

A modification expressed by the word *very* is usually realized by a concentration hedge, i.e., by performing the square of a membership function. However, the

* This work was partly supported by Polish Ministry of Science and Higher Education (Habilitation Project N N516 372234 2008–2011).

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 445–450, 2010.
© Springer-Verlag Berlin Heidelberg 2010

square-hedge operation concentrates γ-membership function (MF) only within its increasing part rather than concentrates the meaning of the words *high* or *low*. The use of the square-hedge *very* we leave to the modification of words like *middle*.

The authors are convinced that hedge operators rather should shift a membership function or extend (or narrow) the segment of this membership function between the empty and the full membership than modify the shape of the original membership function as the classical hedge operators do.

We shall introduce new hedge operators preserving shapes of fuzzy sets. Let A be a fuzzy interval with a membership function $\mu\colon X \rightarrow [0,1]$, i.e. a convex fuzzy subset of X $(\forall u_1, u_2, \lambda \in [0,1],\, f\left(\lambda u_1 + (1-\lambda)\, u_2\right) \geqslant \min\left(f\left(u_1\right), f\left(u_2\right)\right))$ which is also normal $(\exists u \in [0,1] : f\left(u\right) = 1)$. The kernel of A is bounded by m and n, i.e., $[m,n] = [A]_1$. Note that when either m or n reaches the right or the left bound of X, the fuzzy interval is monotone.

The first proposed linguistic operator is the concentration hedge. To preserve an original shape of a membership function, the concentration needs a coefficient scaling slopes of the membership function. For a chosen degree of concentration $c \in (-1,1)$, we look for a coefficient such that $\psi\left(c\right) = 1/\psi\left(-c\right)$, and $\lim_{c\rightarrow -1^+} \psi\left(c\right) = \infty$ for concentration, and $\lim_{c\rightarrow 1^-} \psi\left(c\right) = 0$ for dilation. Formula $\psi = \left(1-c\right) / \left(1+c\right)$ fulfill our requirements. This together with points of application of the concentration, $\mu\left(m\right)$ and $\mu\left(n\right)$, lead us to the following concentration operator:

$$con\left(\mu\left(x\right), c\right) = \begin{cases} \mu\left(\left(x-m\right)\frac{1-c}{1+c} + m\right) & \text{if } x < m \\ \mu\left(\left(x-n\right)\frac{1-c}{1+c} + n\right) & \text{if } x > n \\ 1 & \text{otherwise} \end{cases} . \quad (1)$$

Note that *con* realizes either concentration of the fuzzy set for positive c or dilation for negative c. The concentration hedge acts like dilation semantic modifiers: *relatively*, *somehow* or *more or less*, and concentration semantic modifiers, as *absolutely*.

The second proposed linguistic operator is a shifting hedge. This hedge simply shifts a membership function along its domain X leaving the kernel never compressed. The move is proportional to a degree of intensity. Hence, the shifting operator can be defined as follows:

$$shift\left(\mu\left(x\right), d\right) = \begin{cases} \mu\left(x - d\left(\sup X - n\right)\right) & \text{if } d \geq 0 \\ \mu\left(x - d\left(m - \inf X\right)\right) & \text{otherwise} \end{cases}, \quad (2)$$

where $\sup X$ and $\inf X$ denote the right and left bound of the universe of discourse, respectively.

In the context of monotone fuzzy intervals, the shifting together with a modified word are expressed by linguistic terms like: *extremely high*, *very high*, *more than high*, (*moderately* or *just*) *high*, *more than low*, and *not low*. Note that the use of the comparative description *more than low* and the negative expressions *not low* are dictated by a human way of understanding English words, while the term *not very high* means *high with exception of very high*, and this should be realized by a convex fuzzy set. Observe also that shifting is inseparable with the popular word *very*, although in certain contexts the meaning of *very* should be composed of two operations: shifting and either concentration or dilation, as for the *very high temperature* of a patient.

In the context of two slope fuzzy intervals, the shifting acts in linguistic terms like: *more than usual*, *much more than usual*, or *less than usual*.

In order to transform a single-expert linguistic value into a fuzzy set, we propose to use semantic modifiers which are combinations of the both proposed operators.

3 Interval Type-2 Decoding

Type-1 fuzzy rules of the fuzzy logic system need to be transformed into an intelligible linguistic values with the use of a table coding type-1 fuzzy sets (T1FSs) to linguistic terms. The route of this transformation needs to compare the similarity of the output T1FS and the T2FS from the table. The most similar set found in the table can be directly mapped into its linguistic value by adding proposed hedge modifiers.

As the classic methods for measuring the similarity of interval T2FSs (IT2FSs), the following measures can be counted: a degree of compatibility of Gorzalczany [17], a normal interval-valued similarity measure of Bustince [18], a similarity measure of Mitchell [19]. Recently, Wu and Mendel proposed a two-element Vector Similarity Measure for expressing the similarity of IT2FSs both in shape and proximity [20]. Since from numerical data, we obtain type-1 fuzzy sets (rules) and decode them by means type-2 codebook, an inclusion measure is adequate to the codebook. The inclusion of a type-1 fuzzy set A in a type-2 fuzzy set $\widetilde{B}$ is given as follows:

$$\iota\left(A,B\right)=\frac{\int_{x\in X}\left(\frac{\underline{\mu}_{\tilde{B}}(x)+\overline{\mu}_{\tilde{B}}(x)}{2}\right)\mu_A\left(x\right)dx}{\int_{x\in X}\mu_A\left(x\right)dx}.$$

Using IT2FSs as a symmetric Gaussian function with the center $(\underline{p}_1,\overline{p}_1)$ and width $(\underline{p}_2,\overline{p}_2)$, and using (2), we obtain shift parameter d in the form

$$d=\begin{cases}\frac{p_1-p_1^{'}}{\sup X-p_1^{'}} & \text{if } p_1\geq p_1^{'}\\ \frac{p_1-p_1^{'}}{p_1^{'}-\inf X} & \text{otherwise}\end{cases},\qquad p_1^{'}=\frac{\underline{p}_1^{'}+\overline{p}_1^{'}}{2},\tag{3}$$

where $\sup X$ and $\inf X$ denote the right and the left bound of the universe of discourse, respectively.

In a similar way, using (1), we obtain concentration/dilation parameter c as follows

$$c=\frac{\left(1-p_2^{'}/p_2\right)}{\left(1+p_2^{'}/p_2\right)},\qquad p_2^{'}=\frac{\underline{p}_2^{'}+\overline{p}_2^{'}}{2}.\tag{4}$$

Table 1. Exemplary type-2 codebook generated for the Iris dataset

Domain	low $\left((\underline{p}_1,\overline{p}_1),(\underline{p}_2,\overline{p}_2)\right)$	average $\left((\underline{p}_1,\overline{p}_1),(\underline{p}_2,\overline{p}_2)\right)$	high $\left((\underline{p}_1,\overline{p}_1),(\underline{p}_2,\overline{p}_2)\right)$
sepal length	$((0.71, 0.79), (4.86, 4.94))$	$((0.71, 0.79), (6.06, 6.14))$	$((0.71, 0.79), (7.26, 7.34))$
sepal width	$((0.48, 0.52), (2.38, 2.42))$	$((0.48, 0.52), (3.18, 3.22))$	$((0.48, 0.52), (3.98, 4.02))$
petal length	$((1.17, 1.29), (1.92, 2.04))$	$((1.17, 1.29), (3.89, 4.01))$	$((1.17, 1.29), (5.86, 5.98))$
petal width	$((0.48, 0.52), (0.48, 0.52))$	$((0.48, 0.52), (1.28, 1.32))$	$((0.48, 0.52), (2.08, 2.12))$

Table 2. Parameters of membership functions of type-1 fuzzy rules generated for the Iris dataset

Rule	sepal length(p_1, p_2)	sepal width (p_1, p_2)	petal length(p_1, p_2)	petal width (p_1, p_2)
1	$(0.42, 5.00)$	$(0.46, 3.41)$	$(0.20, 1.47)$	$(0.12, 0.22)$
2	$(0.62, 5.84)$	$(0.38, 2.79)$	$(0.56, 4.25)$	$(0.23, 1.32)$
3	$(0.68, 6.20)$	$(0.38, 2.88)$	$(0.55, 4.41)$	$(0.24, 1.39)$
4	$(0.77, 6.63)$	$(0.38, 2.97)$	$(0.65, 5.50)$	$(0.33, 2.06)$
5	$(0.78, 6.38)$	$(0.38, 2.90)$	$(0.65, 5.39)$	$(0.33, 1.98)$

Table 3. Parameters of linguistic hedges for type-2 fuzzy sets decoded by exemplary type-2 codebook

fuzzy rule	class	sepal length	sepal width	petal length	petal width

r_1	1	somehow(-0.29)
		more(0.03)
		low
			somehow(-0.04)
			more(0.17)
			average
				somehow(-0.72)
				less(-0.52)
				low
					somehow(-0.62)
					less(-0.70)
					low

r_2	2	somehow(-0.10)
		less(-0.14)
		average
			somehow(-0.14)
			more(0.20)
			low
				somehow(-0.37)
				more(0.10)
				average
					somehow(-0.36)
					more(0.02)
					average

r_3	2	somehow(-0.05)
		more(0.05)
		average
			somehow(-0.13)
			less(-0.27)
			average
				somehow(-0.38)
				more(0.16)
				average
					somehow(-0.36)
					more(0.07)
					average

r_4	3	absolutely(0.01)
		more(0.29)
		average
			somehow(-0.14)
			less(-0.19)
			average
				somehow(-0.31)
				less(-0.09)
				high
					somehow(-0.20)
					less(-0.02)
					high

r_5	3	absolutely(0.02)
		more(0.15)
		average
			somehow(-0.14)
			less(-0.25)
			average
				somehow(-0.31)
				less(-0.11)
				high
					somehow(-0.20)
					less(-0.06)
					high

4 Simulation

To test our methodology, we chose Iris dataset as one of the standard classification benchmarks. For this dataset, all input attributes: sepal length, sepal width, petal length and petal width are given in centimeters.

A type-1 fuzzy logic system (algebraic product, singleton consequents) was obtained by the modified density algorithm [21,22,23,24], and is presented in table 2. Then, the rules were decoded according to the type-2 codebook using linguistic hedges described in table 3. Table 1 presents an exemplary type-2

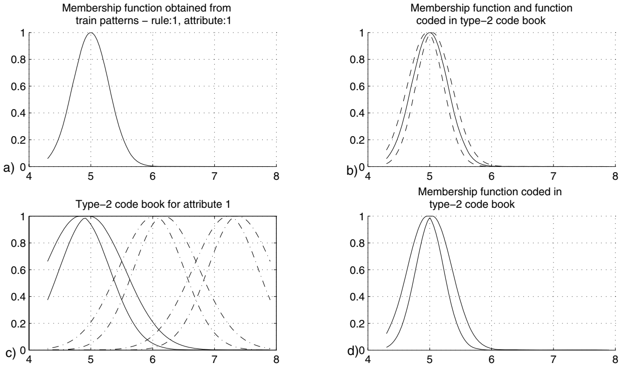


Fig. 1. Coding membership function (a) using type-2 codebook (c)

codebook for Iris data set. The way of coding type-1 fuzzy rules in the type-2 codebook is illustrated in figure 1. The T2FLS, obtained this way, gave not worse performance with a possibility of linguistic interpretations of rules, and it may be a matter of linguistic verification by experts.

5 Conclusion

The aim of this paper was to present a possibility of linguistic interpretation of type-1 fuzzy rules by means of a type-2 codebook. Such interpreted system can be a matter of a linguistic verification by experts. Moreover, if the type-2 codebook contains additional information obtained form experts or other numerical data, information extracted from this codebook can be employed to increase performance of the obtained system.

References

1. Zadeh, L.: The concept of a linguistic variable and its application to approximate reasoning — I. Information Sciences 8, 199–249 (1975)
2. Mendel, J.: Computing with words, when words can mean different things to different people. In: Proceedings of the International ICSC Congress on Computational Intelligence Methods and Applications (1999)
3. Kacprzyk, J., Wilbik, A., Zadrozny, S.: Linguistic summaries of time series via a quantifier based aggregation using the sugeno integral. In: Proc. IEEE-FUZZ 2006, Vancouver, BC, pp. 3610–3616 (2006)
4. Kacprzyk, J., Yager, R.: Linguistic summaries of data using fuzzy logic. International Journal of General Systems 30, 33–154 (2001)
5. Kacprzyk, J., Yager, R., Zadrozny, S.: A fuzzy logic based approach to linguistic summaries of databases. International Journal of Applied Mathematics and Computer Science 10, 813–834 (2000)
6. Kacprzyk, J., Zadrozny, S.: Linguistic database summaries and their protoforms: toward natural language based knowledge discovery tools. Information Sciences 173, 281–304 (2005)

7. Liu, F., Mendel, J.: Encoding words into interval type-2 fuzzy sets using an interval approach. IEEE Transactions on Fuzzy Systems 16(6), 1503–1521 (2008)
8. Lawry, J.: An alternative to compuing with words. Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems 9, 3–16 (2001)
9. Mendel, J.: Computing with words and its relationships with fuzzistics. Information Sciences 177(4), 988–1006 (2007)
10. Mendel, J.: Computing with words: Zadeh, turing, popper and occam. IEEE Computational Intelligence Magazine 2, 10–17 (2007)
11. Niewiadomski, A.: A type-2 fuzzy approach to linguistic summarization of data. Transactions on Fuzzy Systems 16(1) (2008)
12. Türksen, I., Resconi, G.: Fuzzy truthoods based on an additive semantic measure with break of global symmetry in modal logic. International Journal of Fuzzy Systems 8(1), 14–38 (2006)
13. Wang, P. (ed.): Computing With Words. John Wiley & Sons, Inc., New York (2001)
14. Wu, D., Mendel, J.: Aggregation using the linguistic weighted average and interval type-2 fuzzy sets. IEEE Transactions on Fuzzy Systems 15(6), 1145–1161 (2007)
15. Zadeh, L.: From computing with numbers to computing with words from manipulation of measurements to manipulation of perceptions. IEEE Trans. Circuits Syst. I: Fundam. Theory Appl. 4, 105–119 (1999)
16. Zadeh, L.: Fuzzy logic = computing with words. IEEE Transactions on Fuzzy Systems 4, 103–111 (1996)
17. Gorzalczany, M.: A method of inference in approximate reasoning based on interval-valued fuzzy sets. Fuzzy Sets and Systems 21, 1–17 (1987)
18. Bustince, H.: Indicator of inclusion grade for interval-valued fuzzy sets. application to approximate reasoning based on interval-valued fuzzy sets. International Journal of Approximate Reasoning 23(3), 137–209 (2000)
19. Mitchell, H.: Pattern recognition using type-ii fuzzy sets. Information Sciences 170, 409–418 (2005)
20. Wu, D., Mendel, J.M.: A vector similarity measure for linguistic approximation: Interval type-2 and type-1 fuzzy sets. Information Sciences 178, 381–402 (2008)
21. Chiu, S.: Fuzzy model identification based on cluster estimation. J. Intell. Fuzzy Systems 2(3), 267–278 (1994)
22. Tao, C.W.: Unsupervised fuzzy clustering with multi-center clusters. Fuzzy Sets and Systems (128), 305–322 (2002)
23. Dziwiński, P., Rutkowska, D.: Algorithm for generating fuzzy rules for WWW document classification. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 1111–1119. Springer, Heidelberg (2006)
24. Dziwiński, P., Rutkowska, D.: Hybrid algorithm for constructing DR-FIS to classification WWW documents. In: Some New Ideas and Research Results in Computer Science, pp. 105–120. Academic Publishing House EXIT, Warszawa (2006)

Construction of Intelligent Lighting System Providing Desired Illuminance Distributions in Actual Office Environment

Fumiya Kaku[1], Mitsunori Miki[2], Tomoyuki Hiroyasu[3], Masato Yoshimi[2], Shingo Tanaka[1], Takeshi Nishida[1], Naoto Kida[1], Masatoshi Akita[1], Junichi Tanisawa[4], and Tatsuo Nishimoto[4]

[1] Graduate School of Engineering, Doshisha Univ.
[2] Department of Science and Engineering, Doshisha Univ.
[3] Department of Life and Medical Sciences, Doshisha Univ.
[4] Mitsubishi Estate Co., Ltd.
1-3 Tatara Miyakodani Kyotanabe-shi, Kyoto, 610-0321, Japan
{fkaku@mikilab,mmiki@mail,tomo@is,myoshimi@mikilab,stanaka@mikilab
tnishida@mikilab,nkida@mikilab,makita@mikilab}.doshisha.ac.jp,
{junichi_tanisawa,tatsuo_nishimoto}@mec.co.jp

Abstract. An Intelligent Lighting System providing desired illuminance distributions at minimum electrical power was constructed in the actual office environment. The place of construction is the Area Planning Office in the headquarters building of Mitsubishi Estate Co., Ltd. in Otemachi, Tokyo. The floor area is about 240 square meters, and 26 lighting fixtures and 22 illuminance sensors are installed. One lighting fixture consists of neutral fluorescent lamp and light bulb fluorescent lamp, and the color temperature can be changed for each fixture. These devices are connected with the control PC and operate with an optimization algorithm. With the constructed system, individual illuminance was successfully provided to each office worker.

Keywords: Lighting Control, Optimization, System Construction, Illuminance, Office Environment.

1 Introduction

As electronic control technologies and information processing technologies develop in recent years, intelligence has been incorporated in various devices and systems including electric appliances and automobiles, where the system autonomously controls its movement and management depending on a user or environment and burdens to people are reduced. The intelligent system has the ability to take proper actions by thinking, understanding and making judgments based on its own knowledge obtained from information using a sensor, etc. With the intelligent system, autonomous movements in response to the surrounding environment are possible, user satisfaction is improved, and it is possible to flexibly respond to various environmental changes[1].

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 451–460, 2010.
© Springer-Verlag Berlin Heidelberg 2010

Intelligence in lighting systems also began to progress from the viewpoints of realization of illumination patterns that respond to various user requirements as well as of reduction in power consumption. One example is the self control system[2]. With the self control system, the influence by reflected light as well as by daylight is measured with the illuminance sensor built in a light, depending on which the brightness of a lighting fixture is controlled. With this system, it is possible to maintain brightness at a constant level on the desk surface in the measured area, to refrain brightness more than intended at the time of design, and to realize power savings. However, this kind of system controls lights by segment by using fixed illuminance sensors and lights can only be controlled by segment unit; therefore it is not easy to provide discretionary illuminance at a discretionary place.

On the other hand, it is clarified in the study by Boyce, etc. that to provide illuminance most suitable for execution of work for each individual is effective from the viewpoint of improving the lighting environment[3]. To provide brightness most suitable for execution of work for each individual is easily realized with task and ambient lighting. However, ceiling lighting fixtures which provide even brightness on a floor are common in office buildings in Japan, and it is not easy to adopt task and ambient lighting. Therefore, the lighting control system to provide brightness most suitable for each office worker is necessary by using ceiling lighting fixtures.

Based on the above viewpoints, an intelligent lighting system is proposed by the authors. The intelligent lighting system consists of lighting fixtures equipped with a microprocessor, illuminance sensors and an energy meter connected to the network. By controlling each light based on the optimization algorithm that determines luminance intensity, brightness required by a user is provided to a discretionary place. Verification experiments for this system were conducted in the laboratory, which clarified that it is possible to provide required illuminance at several different places under the environment of 15 lights. However, there are far more lights as well as places requiring illuminance in actual offices. For this reason, it is necessary to conduct this kind of large-scale verification experiments upon practical application of the intelligent lighting system. Therefore, the intelligent lighting system was constructed in the actual office in Tokyo, in order to verify the possibility to provide required illuminance.

2 Construction of Intelligent Lighting System in Actual Office Environment

2.1 Purpose of System and Environment for Construction

As described earlier, it is necessary to conduct large-scale verification experiments in an actual office upon practical application of the intelligent lighting system. For this purpose, the intelligent lighting system was constructed in an actual office. The intelligent lighting system constructed is referred to as the "system."

The place of construction is the Area Planning Office in the Otemachi Building (Chiyoda City, Tokyo) owned by Mitsubishi Estate Co., Ltd. In the Area Planning Office, 22 office workers are working on the 16m×15m floor. Each office worker is provided a fixed seat, and one illuminance sensor is installed on each of their desk surfaces. As light source, 26 lighting fixtures consisting of neutral fluorescent lamp (color temperature: 5000K) and one light bulb fluorescent lamp (color temperature: 3000K) manufactured by Mitsubishi Electric Co., Ltd. are installed. The lighting layout and illuminance sensor layout in the Area Planning Office is indicated in Fig. 1.

Under the environment described in the above, the environment to control lights simply by setting illuminance and color temperature required by a user was constructed.

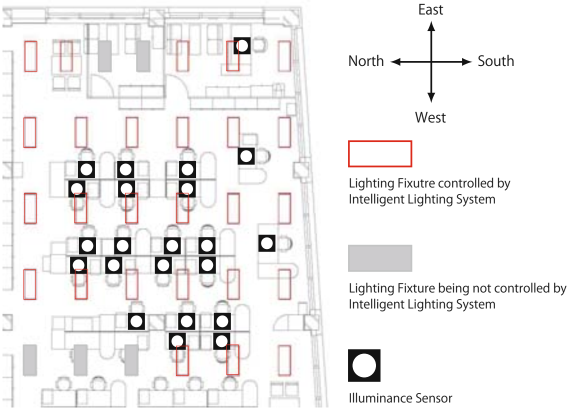

East
North
South
West
Lighting Fixutre controlled by Intelligent Lighting System
Lighting Fixture being not controlled by Intelligent Lighting System
Illuminance Sensor

Fig. 1. Environment for construction

2.2 Overview of the Constructed System

The system provides illuminance required by each office worker (target illuminance) at minimum power consumption with the optimization algorithm based on illuminance and power consumption for each illuminance sensor. It is also possible with the system to individually change color temperature by changing the ratio of luminance intensity for neutral fluorescent lamps and light bulb fluorescent lamps comprising each lighting fixture. However, the color temperature is not controlled.

The target illuminance is established for each illuminance sensor possessed by each office worker. It means that each lighting fixture is controlled so that brightness around the illuminance sensor reaches the target illuminance. Each lighting fixture consists of neutral fluorescent lamp and light bulb fluorescent lamps, and the target illuminance is achieved by controlling the sum of their luminance intensity. The sum of luminance intensity for neutral fluorescent lamps and light bulb fluorescent lamps is referred to as thegluminance intensity for the lighting fixturehand the ratio of luminance intensity as the “gratio of illumination.”

2.3 Composition of the Constructed System

The hardware composition for the system includes one control PC, 26 lighting fixtures (neutral fluorescent lamp and light bulb fluorescent lamp), 3 dimmers (10 channels are mounted on each dimmer), 22 illuminance sensors, and two A/D converters. Connection of the above device is indicated in Fig. 2.

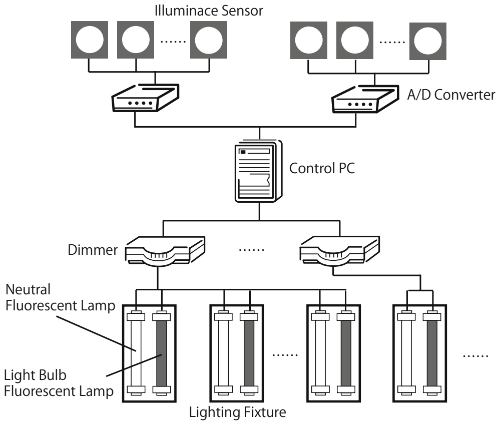


Fig. 2. The hardware composition for the system

It is common in Japan to control the luminance intensity of fluorescent lamps with the Pulse Width Modulation (PWM) method. Therefore, each lighting fixture was connected with a dimmer in order to control luminance intensity for each lighting fixture. The dimmer changes the duty ratio for pulse waves to 256 steps based on the PWM method and sends it to each lighting fixture. Lighting becomes brighter when a pulse wave with a high duty ratio is sent, and darker when a pulse wave with a low duty ratio is sent.

Therefore, it is possible to control luminance intensity for each lighting fixture from a control PC by connecting the control PC with the dimmer. Since a dimmer is able to change a duty ratio independently for each channel, it is possible to independently control the luminance intensity for each light.

As described earlier, information on illuminance and power for each illuminance sensor is necessary to control the intelligent lighting system. Each illuminance sensor was connected to the control PC in order to obtain information on illuminance. However, since information on illuminance from the illuminance sensor used is output by analog signals, they are converted to digital signals through the A/D converter then sent to the control PC.

On the other hand, a device that can obtain real-time information on power over the network is not available; therefore such information is presumed based on the sum of luminance intensity for each light since it is in a proportional relationship with electric energy. Since the luminance intensity for each light is controlled by the control PC, it is possible to operate the system more effectively than using the electric meter. It is noted however that electric energy can be estimated more accurately by adding electric energy for all fluorescent lamps presumed in accordance with the calibration curve for luminance intensity and electric energy for each lamp. However, accurate electric energy is not always necessary at the time of optimization for the purpose of minimizing power consumption.

2.4 Control of the Constructed System

Control Method. In the system, the algorithm where Simulated Annealing (SA) is improved for lighting control (Adaptive Neighborhood Algorithm using Regression Coefficient: ANA/RC) is used to control luminance intensity for each lighting fixture[5].

SA is the algorithm to obtain the optimal solution by randomly generating the subsequent solution near the present solution, to receive the solution depending on the change in the objective function value as well as on the temperature parameter, and to repeat the transitioning processing. However, using SA is not easy for systems being always necessary to respond to an environmental changes, because SA uses the temperature parameter and cooling method. Then, ANA/RC is proposed. ANA/RC obtains the optimal solution by using a variable neighborhood method without using the temperature parameter that is proposed.

It is possible with ANA/RC to provide the target illuminance with minimum power consumption by making luminance intensity for lighting fixtures the design variable and by using the difference between the current illuminance and target illuminance as well as power consumption as objective functions. Furthermore, by learning the influence of each lighting fixture on each illuminance sensor using the regression analysis and by changing the luminance intensity depending on the results, it is possible to promptly change to the optimal luminance intensity. This algorithm is effective to solve the problem which the objective function is near monomodal function and changes in real time.

The luminance intensity for each lighting fixture obtained from the above processing is distributed at the ratio of illumination as the luminance intensity for neutral fluorescent lamps and light bulb fluorescent lamps in accordance with the established color temperature. This ratio of illumination is calculated based on the preliminary experiment where the color temperature is measured by changing the luminance intensity for neutral fluorescent lamps and light bulb fluorescent lamps at a constant level. The ratio of illumination is indicated in Fig. 3 as the proportion of the luminance intensity for neutral fluorescent lamps against the luminance intensity for lighting fixtures. In Fig. 3, the ratio of illumination is not plotted from 3000K to 3300K and from 4600K to 5000K. This is because the dimming range for each fluorescent lamp is limited and the ratio of illumination based on the color temperature in the above cannot be achieved.

With the above processing, it is possible to achieve the target illuminance with the established color temperature.

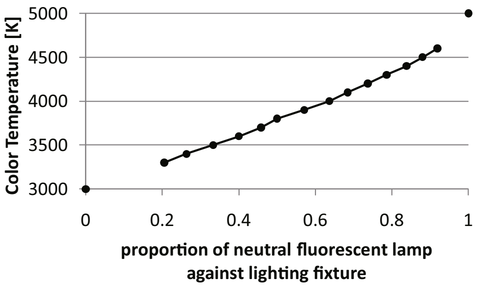


Fig. 3. Change in color temperature

Understanding of the Influence with the Regression Analysis. To understand the influence of each lighting fixture to each illuminance sensor is important to shorten the search time until achieving the target illuminance, because each lighting fixture is able to change the luminance intensity depending on adjacent illuminance sensors by understanding the influence on each illuminance sensor. With ANA/RC, therefore, the regression analysis is conducted to understand the influence of each lighting fixture on each illuminance sensor. As a result of this, the influence can be quantified as the regression coefficient.

Control Flow. The flow of processing in ANA/RC is indicated in the following:

1. Establish initial parameters including initial luminance intensity.
2. Illuminate each light at the initial luminance intensity.
3. Obtain information on illuminance from each illuminance sensor.
4. Estimate power consumption based on the luminance intensity for each light.
5. Calculate the objective function value in the current luminance intensity.
6. Determine the proper range where the next luminance intensity is generated (proximity or neighborhood) based on the regression coefficient.
7. Randomly generate the next luminance intensity within the neighborhood of (6) and illuminate the lighting fixture at the next luminance intensity.
8. Obtain information on illuminance from each illuminance sensor.
9. Estimate power consumption based on luminance intensity for each light.
10. Calculate the objective function value in the next luminance intensity.
11. Conduct regression analysis based on the amount of change in the luminance intensity for the lighting as well as on the amount of change in illuminance for illuminance sensors.
12. Accept the next luminance intensity if the objective function value turns good. If not, return to the previous luminance intensity.
13. Return to (3).

By making the above (3) to (12) as the first step of searching (approximately two seconds) and by repeating this processing, the influence of each light to each illuminance sensor is understood, achieving the target illuminance only with necessary lighting.

Let's discuss on the objective function next. The purpose of the system is to achieve the target illuminance while minimizing power consumption. Therefore, these are formulated as the objective function. The objective function is indicated in the Equation 1.

$$f = P + w * \sum_{j=1}^{n} g_j \tag{1}$$

$$P = \sum_{i=1}^{m} Cd_i$$

$$g_j = \begin{cases} 0 & 0 \leq (Lc_j > Lt_j) > 0 \\ R_j(Lc_j - Lt_j)^2 & (Lc_j - Lt_j) \leq 0 \end{cases}$$

$$R_j = \begin{cases} r_j & r_j \leq T \\ 0 & r_j < T \end{cases}$$

nFnumber of Illuminance sensors, mFnumber of Lighting fixtures
wFweight, PFelectric energy, LcFCurrent illminance
LtFTarget illuminance, CdFluminance Intensity
rFregression coefficient, TFThreshold

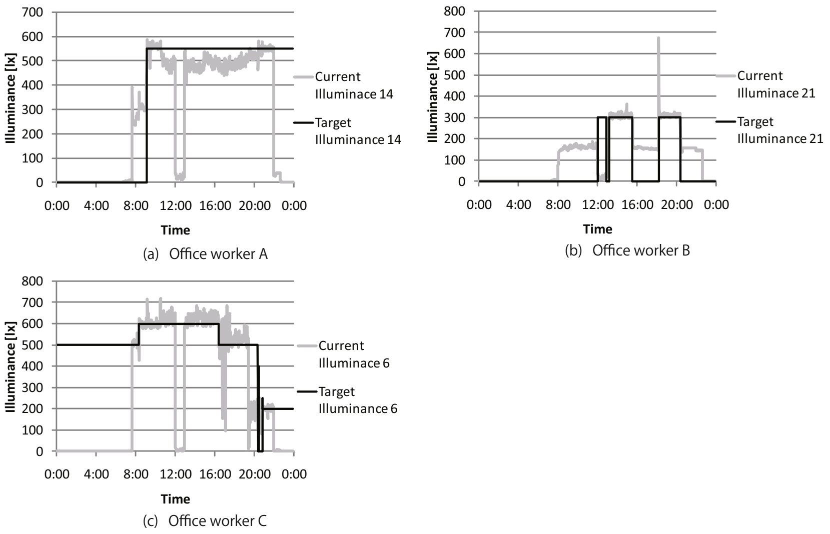
As indicate in the Equation 1, the objective function f consists of power consumption P and constraint g_j. The difference between the current illuminance and target illuminance is used for the constraint g_j, and a penalty is imposed only if the target illuminance is not achieved. As a result, the objective function value largely increases as the target illuminance goes further than the current illuminance. $R_j = 0$ is multiplied if the regression coefficient is less than the threshold. With this, if the illuminance sensor with a lower regression coefficient does not achieve the target illuminance, the objective function value does not increase. Therefore, objects for optimization are successfully limited to illuminance sensors to which the lighting gives a strong influence. Furthermore, the weight w value is multiplied for constraint g_j, and it is possible to switch whether or not to prioritize the convergence to the target illuminance over minimization of power consumption by setting the weight w value.

Let's now discuss on the range where the next luminance intensity is generated (neighborhood). In ANA/RC, the same neighborhood is not used for all lights but multiple neighborhood are properly used depending on situations. Specifically, the neighborhood for the lighting with a high regression coefficient with a sensor that does not achieve the target illuminance is from −1% to 12% of the current luminance (brightening neighborhood). On the contrary, the neighborhood for the lighting with a low regression coefficient with all sensors is from −10% to 1% of the current luminance (dimming neighborhood). In the case that the target illuminance for a sensor with a high regression coefficient is achieved, it is from −2% to 2% of the current luminance (neutral neighborhood). By properly using the three kinds of neighborhood depending on situations, it is possible to promptly achieve the target illuminance while flexibly responding to the change in the environment.

3 Verification of the Operation of the Constructed System

How closely the system was able to provide illuminance compared with the target illuminance required by a user is verified. Upon verification, three out of 22 office workers were extracted, including one who established a rather high target illuminance (Office Worker A), one who established a rather low target illuminance (Office Worker B) and one who regularly changed the target illuminance (Office Worker C). Daily transition of the target illuminance and current illuminance is reviewed.

Fig. 4(a) shows the transition of the target illuminance and current illuminance for the office worker who established the target illuminance at 550 [lx], Fig. 4(b) shows the office worker who established the target illuminance at 300 [lx] and Fig. 4(c) shows the office worker who regularly changed the target illuminance. The vertical axis indicates the illuminance [lx] and the horizontal axis indicates the time.



(a) Office worker A

(b) Office worker B

(c) Office worker C

Fig. 4. Transition of the Target Illuminance and Current Illuminance

There is a timeframe when the target illuminance is set at 0 [lx] in Fig. 4, indicating the situation where the office worker is not at the desk. The timeframes from 0:00 to 7:30, from 12:00 to 13:00, and from 22:30 to 24:00 are indicating values close to 0 [lx] for all current illuminance regardless of the target illuminance. This is because all lights are turned off by the switch, since work is not executed during these timeframes. During the timeframe from 7:30 to 9:00, only the illuminance during daylight from windows is measured.

For the office worker indicated in Fig. 4(a), the target illuminance was set at 550 [lx] at 9:00. It is confirmed that illuminance close to this target illuminance was successfully achieved except during the timeframe when the lights were turned off for lunch break.

The target illuminance was set at 300 [lx] during the timeframes when the office worker was at his desk as indicated in Fig. 4(b). The target illuminance increased from 0 [lx] to 300 [lx] when he returned to his desk at 18:30. At this time, the illuminance reached nearly 700 [lx]. This is because the lights that had been turned off were illuminated again all at once. However, the illuminance then converged to the target illuminance as a result of system control. It is confirmed that illuminance close to the target illuminance was achieved except in the above timeframe. On the other hand, the target illuminance was set at 0 [lx] when the office worker was not at his desk, while the illuminance was not 0 [lx]. This is because necessary lights were illuminated in order to achieve the target illuminance for adjacent office workers.

The office worker indicated in Fig. 4(c) set various target illuminance depending on the time. In this case as well, it is confirmed that illuminance close to the target illuminance was achieved except the timeframe from 16:30 to 17:00. The target illuminance was also set at 500 [lx] and 200 [lx] in early mornings and late evenings, because the office worker forgot to set the target illuminance at 0 [lx] when he left his desk. The current illuminance largely fell below the target illuminance during the timeframe from 16:30 to 17:00, because the receiver of illuminance sensor was covered by documents, etc. and illuminance might have not been properly obtained.

In accordance with the above results, it was confirmed that the system properly controls each lighting fixture to various requirements from office workers.

4 Conclusion

It was discussed in this paper to achieve illuminance required by each office worker in the large-scale intelligent lighting system constructed in the actual office environment. The system operation was then verified based on the log data output for verification experiments. As a result of verification, it was confirmed that the system was successfully controlled in accordance with the target illuminance required by each office worker.

It is considered to be important to review illuminance favored by each office worker and verify long-term energy saving effects by analyzing the log data obtained through long-term operating experiments of the system in the future.

References

1. Miki, M.: Design of intelligent artifacts: a fundamental aspects. In: Proc. JSME International Symposium on Optimization and Innovative Design (OPID 1997), pp. 1701–1707 (1997)
2. Panasonic's Room Light Automatically Changes Brightness to Save Power, http://techon.nikkeibp.co.jp/english/NEWS_EN/20090129/164767/

3. Boyce, P.R., Eklund, N.H., Simpson, S.N.: Individual Lighting Control: Task Performance Mood and Illuminance. Journal of the Illuminating Engineering Society, 131–142 (2000)
4. Miki, M., Hiroyasu, T., Imazato, K., Yonezawa, M.: Intelligent Lighting Control using Correlation Coefficient between Luminance and Illuminance. In: Proc. IASTED Intelligent Systems and Control, vol. 497(078), pp. 31–36 (2005)
5. Shingo, T., Miki, M., Hiroyasu, T.: An evolutional optimization algorithm to provide individual illuminance in workplaces. In: 2009 IEEE International Conference on Systems, Man and Cybernetics, p. 941 (2009)

The Theory of Affinities Applied to the Suppliers' Sustainable Management

Anna María Gil Lafuente and Luciano Barcellos de Paula

Faculty of Economics and Business, University of Barcelona,
Av. Diagonal 690, 08034 – Barcelona – Spain
amgil@ub.edu, luciano@isolucoes.com

Abstract. Diverse scientific studies emphasize that the dialogue with stakeholders is one of the most important points in the area of sustainability in the companies. From the Theory of Stakeholders we will try to analyze the corporate sustainability and the process of suppliers' management of a company that owns the code of conduct in accordance with The Ten UN Global Compact Principles. With the accomplishment of an empirical study one tries to know the degree of fulfillment of the code of conduct on the part of the suppliers and the utility of a tool that facilitates decision making by the employer on the issue. To reach the proposed aim we will resort to the Theory of Affinities, across a model who allows the homogeneous group of variables certain levels. We will use basic elements of decision theory, notably the concepts of relation, as the affinities in the families of Moore and its representation through Galois lattices. The results will provide inputs to the theory of affinities bring to the sustainable management of suppliers.

Keywords: supplier management, corporate sustainability, The UN Global Compact, The Theory of Stakeholders, Logic Fuzzy, Theory of Affinities.

1 State of the Art in the Field

The Stakeholder Theory postulates that a company's ability to generate sustainable wealth over time and thus its long-term value is determined by its relations with its stakeholders (Freeman, 1984). According to himself, the stakeholder of a company is (by definition) any group or individual who can affect or is affected by the achievement of organizational goals. For Elkington (1994) the sustainable development in the company, is one that contributes to sustainable development by providing at the same time, economic, social and environmental benefits - the so-called triple bottom line. In agreement with the Green Book (European Commission, 2001) the corporate responsibility can be defined as "the voluntary integration, by companies, social and environmental concerns in their business operations and their relationships with their stakeholders". The authors Hart and Milstein (2003) use the term "corporate sustainability" to refer to the company that creates value at level of strategies and practices to move towards a more sustainable world. For Hart and

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 461–467, 2010.
© Springer-Verlag Berlin Heidelberg 2010

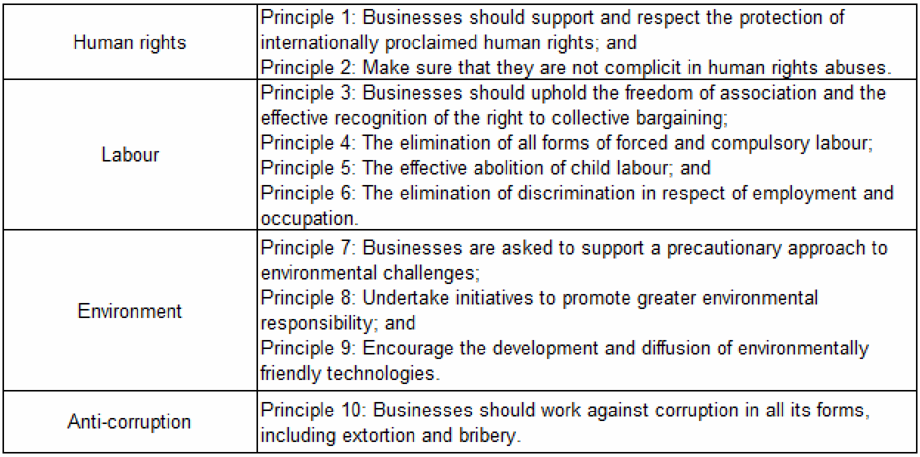
Milstein (2003), sustainability is a complex and multidimensional concept that cannot be solved by a single corporate action. The companies face the challenge of minimizing the residues of the operations in process (the prevention of the pollution), at the same time, the reorientation of your portfolio of skills towards more sustainable technologies and skills (clean technologies). The companies also face the challenge of engaging in extensive interaction and dialogue with external stakeholders in relation to the current offerings (product stewardship), as well as the form in that might develop economically rational solutions to social and environmental problems for the future (vision of sustainability).

2 Intention

The challenge for the companies is to decide what actions and initiatives to continuing and the best way of handling them. We emphasize in our study the management of the suppliers and consider that this one should be a group of essential interest in the search for the sustainability in the companies. Because of the complexity that is to managing suppliers is essential to address the analysis with an approach based on complex systems and models that help entrepreneurs in making decisions, especially in an uncertain environment. For these reasons, it is justified to analyze the management of suppliers using algorithms such as the Theory of Affinities. We will use basic elements of decision theory, principally the concepts of relation. We realize a study of the affinities obtained from the families of Moore and its representation through Galois lattices. We underline some authors who have used the theory of affinities related to personnel management (Gil Aluja, 1987), financial analysis (Gil Lafuente, A.M., 2001), organizational management, commercial management (Gil Lafuente, J., 2001) and sports management (Gil Lafuente, J., 2002). Other studies relate fuzzy logic applied to the sustainability as (Gil Lafuente, A.M. *et al.*, 2005, 2006) in the analysis of the decision of ecological purchase of the consumers, (LU LYY et to., 2007) in the analysis of decision and evaluation of "green" suppliers, (Barcellos Paula; and Gil Lafuente, 2009) process of selection of elements that they contribute to the sustainable growth of the company, and (Barcellos Paula; and Gil Lafuente, 2009) in algorithms applied in the sustainable management of the human resources. On the basis of these precedents the application of the model of affinities to the suppliers' management will allow to choose of efficient form the suppliers according to its practices of sustainable conduct.

3 Used Methodology

According to Gil Aluja (1999) "We define the affinities as those homogeneous groupings occurring at certain levels, and structured in an ordered manner, that link the elements of two distinctive sets, related by the essence of the phenomena that they represent." It is possible to observe the existence of three aspects configurable of the concept of affinity. The first one refers to the fact of which the homogeneity of each group is linked to the level chosen. According to the exigency of each characteristic (elements of one of the sets) a level will be assigned more or less high definer of the threshold from which homogeneity exists. The second expresses the need that the elements of each of the sets are linked together by certain rules of nature in some cases, or by human will on others. The third requires the construction of a structure constituting a certain order that allows the subsequent decision to be taken. The purpose of the group, on the one hand, and the type and strength of the relation between elements of both together, on the other, determines unequivocally all possible groupings. Our study focuses on knowing how a company of the retail sector, which commercializes furniture and objects of decoration, realizes the management of its suppliers with respect to compliance with the code of conduct in accordance with The Ten Global Compact Principles (2008). The empirical study has been realized in August, 2009 by the *Ideas and Solutions Consulting* in Brazil. For request of the contractor, the information of the study was treated by strict confidentiality. According to "The UN Global Compact asks companies to embrace, support and enact, within their sphere of influence, a set of core values in the areas of human rights, labour standards, the environment, and anti-corruption" as presented in Figure 1.

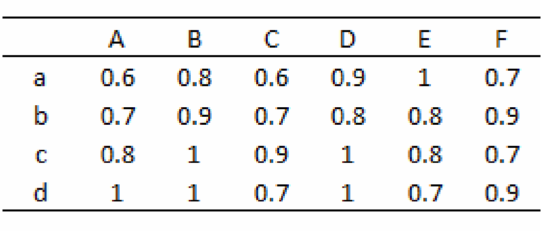


Human rights	Principle 1: Businesses should support and respect the protection of internationally proclaimed human rights; and Principle 2: Make sure that they are not complicit in human rights abuses.
Labour	Principle 3: Businesses should uphold the freedom of association and the effective recognition of the right to collective bargaining; Principle 4: The elimination of all forms of forced and compulsory labour; Principle 5: The effective abolition of child labour; and Principle 6: The elimination of discrimination in respect of employment and occupation.
Environment	Principle 7: Businesses are asked to support a precautionary approach to environmental challenges; Principle 8: Undertake initiatives to promote greater environmental responsibility; and Principle 9: Encourage the development and diffusion of environmentally friendly technologies.
Anti-corruption	Principle 10: Businesses should work against corruption in all its forms, including extortion and bribery.

Fig. 1. The Ten Principles of the United Nations Global Compact

The contracted consultancy takes charge analyzing six suppliers of the company, which will be represented by the set $E^{(1)}=\{A,B,C,D,E,F\}$. The aim is to determine the degree of compliance with the code of conduct for suppliers in their activities according the UN Global Compact. The consultancy realizes an appraisal of the suppliers by means of the scale [0,1], according to which, as the estimation approaches more 1, better it will be the fulfillment of the code of conduct, in the following items: (a) Human Rights , (b) Labour, (c) Environment, (d) Anti-Corruption. We can represent the items of the code of conduct by the set $E^{(2)}$={a,b,c,d}. Following are the results. The valuation would be expressed as follows:

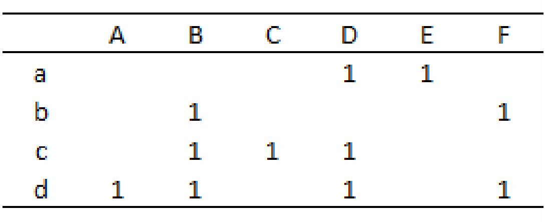
Table 1. Valuation Matrix Suppliers



	A	B	C	D	E	F
a	0.6	0.8	0.6	0.9	1	0.7
b	0.7	0.9	0.7	0.8	0.8	0.9
c	0.8	1	0.9	1	0.8	0.7
d	1	1	0.7	1	0.7	0.9

According to the determination of the company, the level $\alpha \geq 0.9$ is considered to be necessarily to find affinities relations among suppliers and the code of conduct. It implies an exigency very high in the level of fulfillment of the described conducts.

Table 2. Matrix levelffinities $\alpha \geq 0.9$



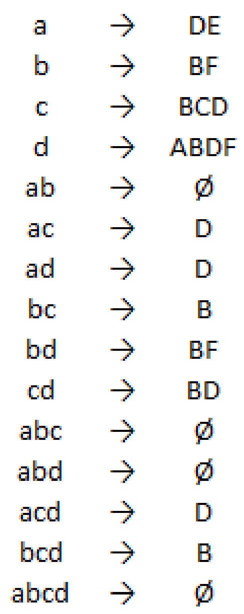
	A	B	C	D	E	F
a				1	1	
b		1				1
c		1	1	1		
d	1	1		1		1

To proceed with the establishment of "relations of affinity" we will use the model called the families of MOORE[1]. We remember that it has been labeled $E^{(1)}$ to the set of suppliers and $E^{(2)}$ to the set of the items of the code of conduct, that is to say, $E^{(1)}$={A,B,C,D,E,F} and $E^{(2)}$={a,b,c,d}. In consequence the major possible set is constituted by all the possible combinations of the elements of the set $E^{(2)}$ that will be:

$P(E^{(2)})$={a,b,c,d,ab,ac,ad,bc,bd,cd,abc,abd,acd,bcd,$E^{(2)}$}.

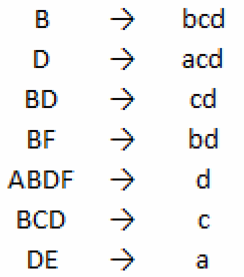
We establish, this way, the existing relations between each element of $P(E^{(2)})$ and the elements of the set $E^{(1)}$. We obtain:

[1] Kaufmann, A.; Gil Aluja, J.: Enterprises management techniques, forecasting, decisions and strategies. 347-405. Ed. Pirámide. Madrid, (1992) (In Spanish).



a	→	DE
b	→	BF
c	→	BCD
d	→	ABDF
ab	→	∅
ac	→	D
ad	→	D
bc	→	B
bd	→	BF
cd	→	BD
abc	→	∅
abd	→	∅
acd	→	D
bcd	→	B
abcd	→	∅

The result will allow to obtain to the level $\alpha \geq 0.9$ the not empty elements of P(E$^{(1)}$) and the subsets of E$^{(2)}$ that are not included in others, it will be had this way:



B	→	bcd
D	→	acd
BD	→	cd
BF	→	bd
ABDF	→	d
BCD	→	c
DE	→	a

Therefore the affinities obtained are as follows: (bcd,B), (acd,D), (cd,BD), (bd,BF), (d,ABDF), (c,BCD), (a,DE). These results can be shown graphically in the following Galois Lattice[2]:

[2] Kaufmann, A.; Gil Aluja, J.: Special techniques for expert management. Milladoiro, Santiago de Compostela, pp. 151-175 (1993) (In Spanish).

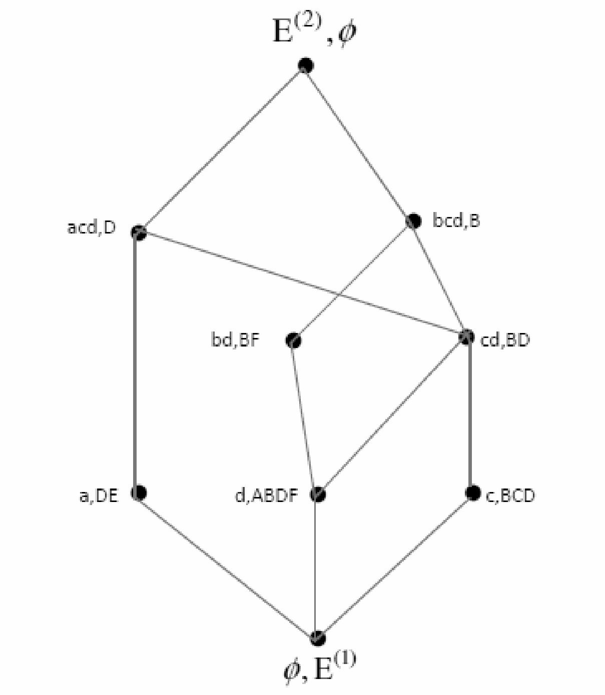


Fig. 2. Galois lattice

4 Obtained Results

This network structure reveals in a visual way the existing affinities among the different suppliers in relation to the contents of the code of conduct of the same ones. We emphasize that the suppliers D and B have reached the best results relating to the general fulfillment of the code of conduct established by the company. The result shows that the supplier D fulfills with the items (*a) Human Rights, (b) Labour, (c) Environment*, and that it needs the topics improve related with (*d) Anti-corruption.* On the other hand, the supplier B fulfills with the items (*b) Labour*, (*c) Environment, (d) Anti-corruption*, and that is necessary to improve the topics related with (a) *Human Rights*. The contribution of this paper is to provide a model that assist employers in decision making and sustainable management of suppliers, be a useful tool to be used in the process of grouping. In addition the model serves to establish relations between different concepts for different levels of fulfillment of the analyzed variables and to obtain the corresponding affinities. The Galois lattice shows visually the similarities existing among the various suppliers in relation to degrees of compliance code of conduct for themselves. We think that our contribution will serve to support future research in the field of sustainability in the companies and the application of the methodology in management with stakeholders.

Acknowledgments. Luciano Barcellos de Paula is as a scholar of MAEC-AECI.

References

1. Barcellos Paula, L., Gil Lafuente, A.M.: Process of selection of elements that they contribute to the sustainable growth of the company. In: International Conference and Doctoral Consortium for ISEOR and Academy of Management held at Lyon, France, vol. 1, pp. 773–788 (2009) (in Spanish)
2. Barcellos Paula, L., Gil Lafuente, A.M.: Algorithms applied in the sustainable management of the human resources. In: XV Congress of International Association for Fuzzy-Set Management and Economy (SIGEF), Lugo, Spain, Lugo, October 29-31 (2009) (in Spanish)
3. European Commission: The Green Book, Promoting a European framework for corporate social responsibility. Green Paper. Luxembourg: Office for Official Publications of the European Communities (2001)
4. Elkington, J.: Towards the Sustainable Corporation: Win-Win-Win Business Strategies for Sustainable Development. California Management Review 36(2), 90–100 (1994)
5. Elkington, J.: Cannibals with forks: the triple bottom line of 21st Century Business. U.K. Capstone Publishing Limited, Oxford (1998)
6. Freeman, R.E.: Strategic Management: A Stakeholder Approach. Pitman Series in Business and Public Policy (1984)
7. Gil Aluja, J.: Recruitment: the problem of versatility and uniformity. In: Cuadernos CEURA, Madrid, pp. 8–10 (1987) (in Spanish)
8. Gil Aluja, J.: Elements for a theory of decision in uncertainty. Vigo, Milladoiro, p. 186 (1999) (in Spanish)
9. Gil Lafuente, J.: Model for the homogeneous grouping of the sales forces. In: Congress, M.S., Changsha (Hunan), R.P. (eds.), China, September 25-27, pp. 332–335 (2001)
10. Gil Lafuente, J.: Algorithms for excellence. Keys to success in sport management. Vigo. Milladoiro (2002) (in Spanish)
11. Gil Lafuente, A.M., Gil Lafuente, J.: Models and Algorithms for the treatment of creativity in business management. Editorial Milladoiro., pp. 47–91 (2007) (in Spanish)
12. Gil Lafuente, A.M., Salgado Beltrán, L.: Models for analysing purchase decision in consumers of ecologic products. Fuzzy Economic Review X, 47–62 (2005)
13. Gil Lafuente, A.M., Salgado Beltrán, L., Subirá Lobera, E., Beltrán, L.F.: Forgotten effects theory in the sustainable consumption of organic products. In: Sustainable Development: Myth or reality?, pp. 223–240. Ed. Centro de investigaciones biológicas del noroeste, S.C. Mexico, pp. 223-240 (2006) (in Spanish)
14. Gil Lafuente, A.M.: New strategies for financial analysis in business. Barcelona, Ariel (2001) (in Spanish)
15. Hart, S.L., Milstein, M.: Creating Sustainable Value. Academy of Management Executive 17(2) (2003)
16. Kaufmann, A., Gil Aluja, J.: Special techniques for expert management. In: Milladoiro, Santiago de Compostela, pp. 151-175 (1993) (in Spanish)
17. Kaufmann, A., Gil Aluja, J.: Enterprises management techniques, forecasting, decisions and strategies. Ed. Pirámide, Madrid, pp. 347–405 (1992) (in Spanish)
18. Lu, L., Wu, C., Kuo, T.: Environmental principles applicable to green supplier evaluation by using multi-objective decision analysis. International Journal of Production Research 45(18-19), 4317–4331 (2007)
19. UN Global Compact: United Nations Global Compact Office, New York City (2008)

Protrace: Effective Recursion Tracing and Debugging Library for Functional Programming Style in Common Lisp

Konrad Grzanek and Andrzej Cader

IT Institute
Academy of Management (SWSPiZ)
Sienkiewicza 9, 90-113 Lodz
kongra@gmail.com, acader@swspiz.pl

Abstract. Among programming styles the functional one has gained an increasing attention in the recent years. Lisp dialects and Common Lisp in particular have always been languages of choice for artificial intelligence systems implementation. Debugging functional algorithms based in the natural way on recursion and immutability is especially hard. This article presents a library that allows the programmer to mark and trace Common Lisp expressions evaluation.

Keywords: Functional Programming, Common Lisp, Artificial Intelligence, Debugging.

1 Introduction

Functional programming style was always regarded one of the most effective ways of building robust, dependable software systems. It was underlined - among the others - by Backus in his famous ACM Turing Award lecture [3]. It's main advantage comes from the lack of assignment and all harmful features of this side-effect operator [9]. The functional style was one of the two major styles, together with the imperative one. Both these styles have deep mathematical roots; the universal Turing machine [1] in the case of imperative paradigm and λ-calculus [2] in the case of the functional.

The ability to express programs by pure function combinations was always and still is a very attractive perspective for solution builders in the Artificial Intelligence domain ([10], [11]). Lisp as specified by McCarthy [8] with it's dialects was actually the first pragmatic approach to solve AI software problems using the functional style of programming.

Common Lisp was the first ANSI standardized Lisp variant [14], [16]. But the popularity of functional style led to some other achievements in the computer languages design and implementation field [4], e.g. Haskell [5] and ML programming languages that are statically typed in the opposition to the dynamic type system of Lisp.

On the other hand, the software industry has been dominated by the imperative style, considered standard, traditional etc. However in the recent years a

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 468–475, 2010.
© Springer-Verlag Berlin Heidelberg 2010

return to the original ideas of Church and McCarthy may be observed. The parallel processing raises severe problems related to mutual resources access when using the imperative approach.

This trend led to some interesting works on the persistent (effectively immutable) data structures [6], [7]. The latter work by Bagwell is a basis of Clojure [18] programming language, which is a Lisp targeting JVM and the Java programming ecosystem by a very close integration with the Java run-time [19].

Also, there is a renaissance in the Common Lisp academic and industrial society. Works by Graham [16], Norvig [11] and Seibel [13] present Common Lisp as a generic tool for solving general-class software problems, not only in AI, but also in a day-to-day programmer's activity.

The urge to popularize the functional style among the programmers raises the need of having good development environments and effective debugging. This article presents a debugging library for Common Lisp designed and developed by the authors.

2 Debugging Common Lisp Programs

There are two general flavors of finding bugs in declarative languages:

1. Semantic debugging whose goal is to find semantic errors. This process requires a very detailed research and it undergoes one. Works on declarative debugging of logic programming systems [20] and lazy functional languages [21] are especially important.
2. Run-time debugging based on observations of expression values on the program execution time. Some great programmers [12] underline the fact that this is still a very important and valuable technique for finding both low- and high-level conceptual and architectural issues.

In this paper we concentrate on the second issue. Our goal was to create a simple algorithms and a library capable of tracing CL expressions evaluation on the program run-time, possibly considering an integration with a CL REPL. We decided to use Steel Bank Common Lisp (SBCL) implementation, but the library being the result of our work is capable of being used with other CL variants.

Our assumptions were:

- Tracing some explicitly indicated expressions would be the debugging technique.
- The mechanisms would be used to trace deep recursive calls.
- All library elements weaved into the debugged code would be transparent to normal evaluation flow. In particular the would not interfere the functional character of the language.

3 Implementation

As usual in functional languages, all CL expressions have values. E.g. the simple expression

```
(+ 1 2)
```

evaluates to the value 3. The idea behind Protrace is to wrap expressions into enhanced forms. A special code-enhancing form is called prot-watch:

```
(prot-watch (+ 1 2))
```

It evaluates directly to the same value 3 as the embedded form (+ 1 2). prot-watch is a macro that gets macro-expanded on the compile time. The final macro-expanded form may be traced like below:

```
(macroexpand-1 '(prot-watch (+ 1 2)))
```

It evaluates to

```
(prot-with-eval-level
  *watch-marker*
  (when *callback* (funcall *callback* '(+ 1 2) '(+ 1 2)
                            *eval-level* nil nil))
  (let ((#:value821 (+ 1 2)))
    (when *callback*
      (funcall *callback* '(+ 1 2) '(+ 1 2) *eval-level*
               #:value821 t))
    #:value821))
```

The initial expression (+ 1 2) gets wrapped into a form that guaranties the following actions taking place during evaluation of the whole form:

1. When a function *callback* is bound, it gets called with the parameters that denote the wrapped form, the display form (if the programmer needs some special visual form of the traced expression), the current evaluation level (stack level) being the actual binding of *eval-level* variable and two NIL values meaning the value of the traced expression is not known yet. It's due to the fact that the evaluation of the original expression (+ 1 2) is yet to be done.
2. The original expression (+ 1 2) is evaluated and it's value is assigned to a temporary symbol (:value821 in this case). The symbol is auto-generated on the compile (macro-expansion) time to avoid variable capture problems [15].
3. *callback* function is called for the second time (again only if it's bound) with the value of the evaluation process of the original expression.
4. Finally the value of the evaluation gets returned as the value of the whole prot-watch form.

Some notes on symbols *callback* and *eval-level*. They are dynamic variables declared in the main package of the Protrace library. Initially they are bound to NIL and 0, so that when the raw form (prot-watch ...) gets evaluated, nothing interesting really happens. Only the value of the embedded form gets returned.

The *eval-level* changes during run-time within (prot-with-eval-level ...) macro-expansion form. This is another syntactic and semantic form that allows to control the stack depth on the run-time. The form is completely side-effect free from the programmer's viewpoint, however it gets expanded into a target form whose only side-effect is to change the *eval-level*. E.g.:

```
(prot-with-eval-elevel expr)
```

has the macro-expansion like:

```
(let ((#:original-level789 *eval-level*))
  (incf *eval-level*)
  (unwind-protect (progn expr) (setf *eval-level*
                                     #:original-level789)))
```

Expressions wrapped in the (prot-with-eval-level ...) form are executed inside a safe environment that keeps track of *eval-level*, increasing it and then restoring the original value. It is then an explicit way to actually monitor some aspect of the stack, it's depth in this case.

Because every (prot-watch ...) form macro-expands into
(prot-with-eval-level ...), it is also automatically increases the *eval-level*.

The most low-level mechanism that actually performs the full evaluation of the traced expression is called prot-trace-expr. It's full signature is:

```
(prot-trace-expr
   expr
   (form-symbol display-form-symbol
    eval-level-symbol
    value-symbol value-p-symbol)
   &body body)
```

Evaluation of each (prot-watch ...) expression covered by the ongoing trace is being reported by calling body of the (prot-trace-expr ...) twice - before and after evaluation, with a set of parameter symbols:

- form-symbol. Depicts the form being evaluated (including all prot-watch clauses around and inside).
- display-form-symbol. Display-form of the given prot-watch if specified in prot-watch expression, or else the form being evaluated stripped of a prot-watch clauses,
- eval-level symbol. Depicts nesting depth of the evaluated form, both inside other traced expressions and in call stack (including recursion).

– value-symbol. Expression value if post-evaluation call, nil if call takes place before evaluation.
– value-p-symbol. True (t) when post-evaluation call, nil otherwise.

This very flexible macro is capable of being a strong basis for other abstraction layers. The form (prot-build-expr-trace . . .) is such a higher-level tracing macro that gets expanded into an algorithm that works on the run-time and builds a tree representing a trace of prot-watched expressions. E.g:

```
(prot-build-expr-trace (+ 1 2))
```

gets expanded into

```
(let* ((#:tree965 (cons (list nil 'trace 0 '"") nil))
       (#:stack966 (list #:tree965)))
  (prot-trace-expr (+ 1 2)
      (#:form968 #:display-form969 #:eval-level970 #:value971
       #:value-p972)
    (if (not #:value-p972)
        (push (cons nil nil) #:stack966)
        (let ((#:node967 (pop #:stack966)))
          (set-node-value #:node967
                          (list #:form968 #:display-form969
                                #:eval-level970
                                #:value971))
          (add-node (car #:stack966) #:node967))))
  #:tree965)
```

Again the form is built upon (prot-trace-expr . . .) - a lower level abstraction layer.

Results produced on the run-time by (prot-build-expr-trace . . .) may be used to present the flow of control in various target forms. We use only the most basic - textual one.

```
(prot-print-expr-trace (prot-watch (+ 1 2)))
```

results in a following output on the CL REPL console:

```
TRACE =>
 (+ 1 2) => 3
```

(prot-print-expr-trace . . .) simply takes the value of (prot-build-expr-trace . . .) and pretty-prints it. It's worth noting that when there are no prot-watched expressions, the output is empty:

```
(prot-print-expr-trace (+ 1 2))
```

```
TRACE =>
```

4 Examples

Protrace may be used to analyze the recursive behavior and recursive procedures calls. E.g. the simple factorial [1]

```
(defun factorial (n)
  (if (= n 0)
      1
      (* n (factorial (1- n)))))
```

may be enhanced by the programmer to the form like

```
(prot-traceable-defun factorial (n)
  (if (= n 0)
      1
      (prot-watch (* (prot-watch n) (factorial (1- n))))))
```

giving a result on the evaluation (run) time

```
(prot-print-expr-trace (factorial 10))
```

```
TRACE =>
  (* N (FACTORIAL (1- N))) => 3628800
   N => 10
    (* N (FACTORIAL (1- N))) => 362880
     N => 9
      (* N (FACTORIAL (1- N))) => 40320
       N => 8
        (* N (FACTORIAL (1- N))) => 5040
         N => 7
          (* N (FACTORIAL (1- N))) => 720
           N => 6
            (* N (FACTORIAL (1- N))) => 120
             N => 5
              (* N (FACTORIAL (1- N))) => 24
               N => 4
                (* N (FACTORIAL (1- N))) => 6
                 N => 3
                  (* N (FACTORIAL (1- N))) => 2
                   N => 2
                    (* N (FACTORIAL (1- N))) => 1
                     N => 1
```

(prot-traceable-defun . . .) is just a shorthand for

[1] This definition does not contain validity check (positive integer) of n for simplicity reasons.

```
(defun <name> (<params>)
  (prot-with-eval-level ...))
```

The defined function factorial may be used without any further modifications outside the (prot-print-expr-trace . . .) form.

Another more complicated use-case is a mystery function puzzle solution. The mystery function puzzle is one of the famous programming puzzles published by ITA Software [22]. The non-trivially recursive function is defined as

```
(defun m (i j k)
 (cond ((= i 0) (1+ k))
        ((and (= i 1) (= k 0)) j)
        ((and (= i 2) (= k 0)) 0)
        ((= k 0) 1)
        (t (m (1- i) j (m i j (1- k))))))
```

Goal is to find (m 4 4 4). The enhanced version of the function has a form like the following

```
(defun m (i j k)
 (prot-watch (list i j k) :display-form "of ")
 (cond ((= i 0) (prot-watch (1+ k)))
       ((and (= i 1) (= k 0)) (prot-watch j))
       ((and (= i 2) (= k 0)) (prot-watch 0))
       ((= k 0) (prot-watch 1))
       (t (prot-watch (m (1- i) j (prot-watch (m i j (1- k))))))
 ))
```

After some observations of how the values of parameters impacted the execution flow, it was possible to transform m into a non-recursive shape and to find the final value even though it didn't fit into the memory of a typical PC machine.

5 Conclusions

The library and language extension implemented by us is a very convenient way to observe recursive behaviors of procedures. It is an interesting alternative to using IDEs and their debuggers. In particular when important processes take place with a very deep recursion, as in the case of the mystery function.

References

1. Turing, A.M.: On computable numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society 42(2) (1936)
2. Church, A.: A set of postulates for the foundation of logic. Annals of Mathematics, Series 2, 33, 346–366 (1932)

3. Backus, J.: Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs. ACM Turing Award Lecture (1977); Communications of the ACM 2 (August 1978)
4. Hudak, P.: Conception, Evolution, and Application of Functional Programming Languages. ACM Computing Surveys 21(3) (September 1989)
5. Peyton Johnes, S.L.: The Implementation of Functional Programming Language. Prentice Hall International (UK) Ltd., Englewood Cliffs (1987)
6. Okasaki, C.: Purely Functional Data Structures. PhD thesis submitted to School of Computer Science Carnegie Mellon University Pittsburgh, PA 15213 (1996)
7. Bagwell, P.: Ideal Hash Trees. Es Grands Champs 1195 (2001)
8. McCarthy, J.: Recursive Functions of Symbolic Expressions and Their Computation by Machine. CACM 3(4), 184–195 (1960)
9. Abelson, H., Sussman, G.J.: Structure and Interpretation of Computer Programs. MIT Press, Cambridge (1984), ISBN 0-262-01077-1
10. Russel, S.J., Norvig, P.: Artificial Intelligence A Modern Approach, 2nd edn. Pearson Education Inc., Upper Saddle River (2003)
11. Norvig, P.: Paradigms of Artificial Intelligence Programming: Case Studies in Common Lisp. Morgan Kaufmann, San Francisco (1991)
12. Seibel, P.: Coders at Work, 1st edn. Apress (September 16, 2009)
13. Seibel, P.: Practical Common Lisp, 1st edn. Apress (April 11, 2005), `http://www.gigamonkeys.com/book/`,
14. Steele, G.L.: Common Lisp the Language, 2nd edn. Digital Press (1990), ISBN 1-55558-041-6, `http://www.cs.cmu.edu/Groups/AI/html/cltl/cltl2.html`
15. Graham, P.: On Lisp - Advanced Techniques for Common Lisp. Prentice-Hall, Englewood Cliffs (1993)
16. Graham, P.: ANSI Common Lisp. Series in Artificial Intelligence. Prentice-Hall, Englewood Cliffs (1996)
17. Steel Bank Common Lisp Website, `http://www.sbcl.org`
18. Clojure Website, `http://clojure.org`
19. Halloway, S.: Programming Clojure. The Pragmatic Bookshelf (May 2009), ISBN: 978-1-93435-633-3
20. Ferrand, G.: Error Diagnosis in Logic Programming, an Adaptation of E. Y. Shapiro's Method. Journal of Logic Programming 4(3), 177–198 (1987)
21. Nilsson, H.: Tracing piece by piece: affordable debugging for lazy functional languages. In: Proceedings of the 1999 ACM SIGPLAN Int'l Conf. on Functional Programming, pp. 36–47. ACM Press, New York (1999)
22. ITA Software Programming Puzzles, `http://www.itasoftware.com/careers/puzzle_archive.html?catid=39#TheMysteryMFunction`

Automatic Data Understanding: A Necessity of Intelligent Communication*

Wladyslaw Homenda

Faculty of Mathematics and Computer Science
University of Bialystok
ul. Sosnowa 64, 15-887 Bialystok, Poland
and
Faculty of Mathematics and Information Science
Warsaw University of Technology
Plac Politechniki 1, 00-660 Warsaw, Poland
homenda@mini.pw.edu.pl

Abstract. In this paper aspects of intelligent man-machine communication are considered. Selected topics are deemed from a perspective of data processing. Human beings expect that data processing is under a kind of intelligent control. This expectation forms natural properties that communication is focused on revealing structures of data being exchanged and on controlling operations to which these structures of data are subjected to. Such qualities are a scanty substitute of the main goal of communication, which is an understanding data being exchanged. An exposition of these topics is illustrated with an example from the domain of music notation. The discussion leads to a conclusion that structuring and understanding of processed data is a basis of intelligent man-machine communication.

Keywords: man-machine communication, information processing, data structuring, data understanding.

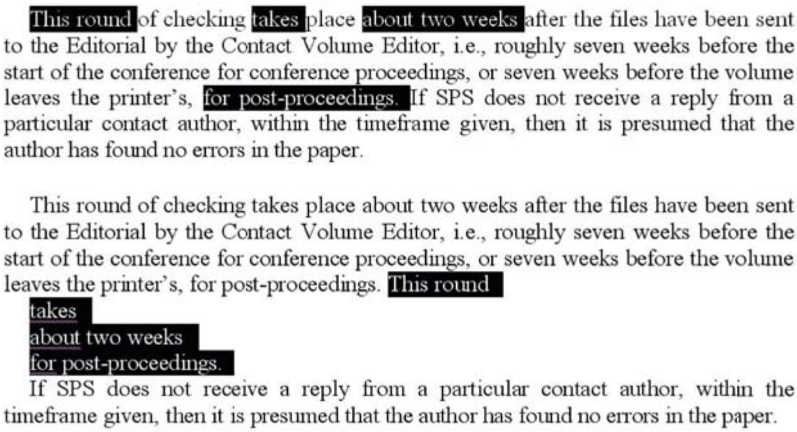
1 Introduction

A fundamental feature of any computer program is its ability to communicate with its environment, i.e. with human users or with other computer programs. Besides standard functionality of computer programs like, e.g. operations on files (New, Open, Save, Print, ...) or operations on Desktop elements (New Window, Cascade, Tile, Close Window, Zoom, ...), communication is related to data processed by the program. Human users' natural expectation, that communication would be focused on mentioned above scanty substitutes of an understanding (of data), is still a wishful thinking rather than reality. An intelligent man-machine and machine-machine communication is a desirable though usually unfeasible attribute of modern computer programs.

* The research is supported by The National Centre for Research and Development, Grant no N R02 0019 06/2009.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 476–483, 2010.

© Springer-Verlag Berlin Heidelberg 2010



This round of checking takes place about two weeks after the files have been sent to the Editorial by the Contact Volume Editor, i.e., roughly seven weeks before the start of the conference for conference proceedings, or seven weeks before the volume leaves the printer's, for post-proceedings. If SPS does not receive a reply from a particular contact author, within the timeframe given, then it is presumed that the author has found no errors in the paper.

This round of checking takes place about two weeks after the files have been sent to the Editorial by the Contact Volume Editor, i.e., roughly seven weeks before the start of the conference for conference proceedings, or seven weeks before the volume leaves the printer's, for post-proceedings. This round
takes
about two weeks
for post-proceedings.
If SPS does not receive a reply from a particular contact author, within the timeframe given, then it is presumed that the author has found no errors in the paper.

Fig. 1. Example of Multiselect, Copy (upper part of the figure) and Paste (bottom part of the figure) operations in Microsoft Word

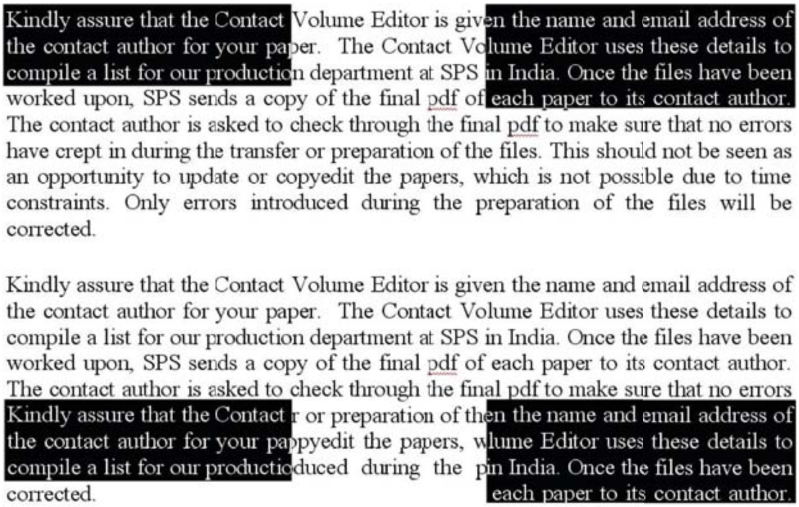
Modern computer programs provide tools allowing operation on processed data. The set of basic operations - Select, Copy and Paste, Find and Replace, Sort, etc. - are available in almost every computer programAll such operations are performed at structures of data. Such structures are usually based on internal data format of given computer program, what affects control over data processing and communication between such a program and a user.

The *Select* operation allows for selection of a data on a screen. This is a fundamental interface operation used in other interface operations. Realization of the *Select* operation in typical computer programs is performed on a plain data without considering a structure of information.

Data processed in text editors is considered as a sequence of text symbols (letters, digits, punctuation marks), words, paragraphs. Selection in a text is usually a continuous part of a text. A selection of several separated parts of a text, so-called multiselect opration, is possible in some editors. However, multiselection requires interpretation in further processing. Interpretation of a form and a structure of a multiselection is rather hard and usually a simple concatenation of all selected parts is employed.

An example of Multiselect, Copy and Paste operations in Microsoft Word is presented in Figure 1. A multiselection is performed as a series of mouse selections with Ctrl key pressed. Selected pieces of the text are copied to the clipboard. They are then pasted as successive lines, i.e. every piece is appended with a line break.

Another example of Multiselect, Copy and Paste operations in Paint Shop Pro is shown in Figure 2. A multiselection is performed as a series of mouse selections with Shift key pressed. Selected parts of an image in a form of rectangles are directly pasted to a target image. Source (pasted) rectangles replace target area.



Kindly assure that the Contact Volume Editor is given the name and email address of
the contact author for your paper. The Contact Volume Editor uses these details to
compile a list for our production department at SPS in India. Once the files have been
worked upon, SPS sends a copy of the final pdf of each paper to its contact author.
The contact author is asked to check through the final pdf to make sure that no errors
have crept in during the transfer or preparation of the files. This should not be seen as
an opportunity to update or copyedit the papers, which is not possible due to time
constraints. Only errors introduced during the preparation of the files will be
corrected.

Kindly assure that the Contact Volume Editor is given the name and email address of
the contact author for your paper. The Contact Volume Editor uses these details to
compile a list for our production department at SPS in India. Once the files have been
worked upon, SPS sends a copy of the final pdf of each paper to its contact author.
The contact author is asked to check through the final pdf to make sure that no errors
Kindly assure that the Contact r or preparation of then the name and email address of
the contact author for your pappyedit the papers, wlume Editor uses these details to
compile a list for our productioduced during the pin India. Once the files have been
corrected. each paper to its contact author.

Fig. 2. Example of Multiselect, Copy (upper part of the figure) and Paste (bottom part of the figure) operations in Paint Shop Pro

The examples of multiselection, copying and pasting are performed at a very low level of data structures. In the text case these operation are executed on sequences of alphanumeric symbols without any concern of such text elements like nouns, noun groups, etc. The operations applied to images are executed at the level of pixels, even if a selection is performed with regard to some condition defined, e.g. selection of pixels of a given color.

An example of Multiselect, Copy and Paste operations done on a structure of information instead on primitive data elements is shown in Figure 3. This example was performed in vector graphics of CorelDraw editor[1]. The example is based on a derivation tree in a context-free grammar, cf.[2]. A multiselection of a tree piece is bolded in the left part of Figure 3. The pasted tree piece is bolded. It replaced the bottom right *empty* leaf. A multiselection chooses several object of vector graphics. In CorelDraw objects laying in a rectangle could be indicated for selection. In this example, such a primitive selection tool allowed for choosing a part of a derivation tree and it created a sensible information structure. The pasting is a recurrent expansion of the bottom right node $< ts\_stave >$ of the tree; it is bolded in the right part of Figure 3. A reasonable pasting requires consistency of a pasting context and a pasted structure. In this example a pasting context is interpreted as such a node of the derivation tree, which provide consistency of its expansion with the grammar corresponding to this tree. However, the program provides no tool for finding such a node. User of the program must do it.

[1] Note: usual difference between raster and vector graphics as, for instance, data compression or quality of zoomed images, is not a subject of interest here.

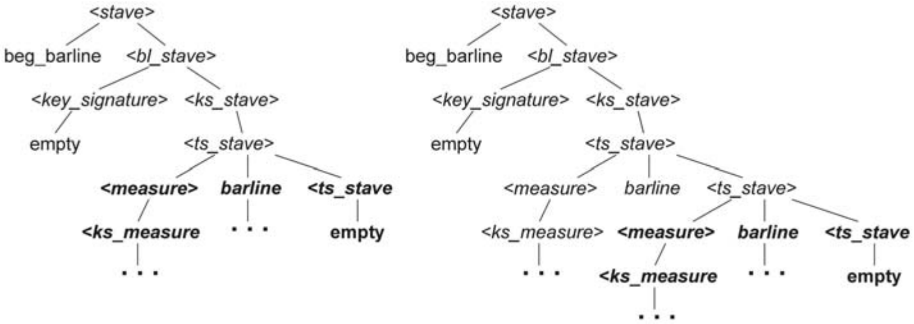


Fig. 3. Example of Multiselect, Copy (left part of the figure) and Paste (right part of the figure) operations in CorelDraw

Find and Replace is a pair of structural operations based on selection. Find operation is a kind of selection of a data equal to given data (this given data is usually selected prior to Find operation or is defined in a similar way). Data comparing is the fundamental part of Find operation. In case of text processors strings of symbols are compared. In editors of raster graphic comparing has little sense. In editors of vector graphics comparing may be performed on a set of objects' primitives like object types (e.g. arcs, rectangles, ellipse), background color, background pattern and texture. But this is not what is usually meant by intelligent man-machine communication.

2 Structural Operations in the Space of Music Information

A language of music notation is structured by its nature. Even a small operation, e.g. insertion or deletion of a note, may substantially change a structure of information affecting voices in the measure of insertion/deletion. In this section we consider several important structural operations on a space of music information: music notation recognition, visualization of recognized information, selection, transposition, separation of voices, conversion between different formats of music representation. Such operations could be performed in a format of music representation carefully designed for them. These operations illustrate music processing which could be rightly called intelligent. They are intelligent in a sense that they follow a man doing them, cf. [4]. Execution of such operations involves music knowledge as well as patience needed for repeating the same action many times (e.g. transposition of a voice involves shifting dozens of notes and placing them in a correct configuration).

2.1 Music Notation Recognition and Representation

Automatic music notation recognition and representation of recognized information is a fundamental structural operation. This operation is an example of a

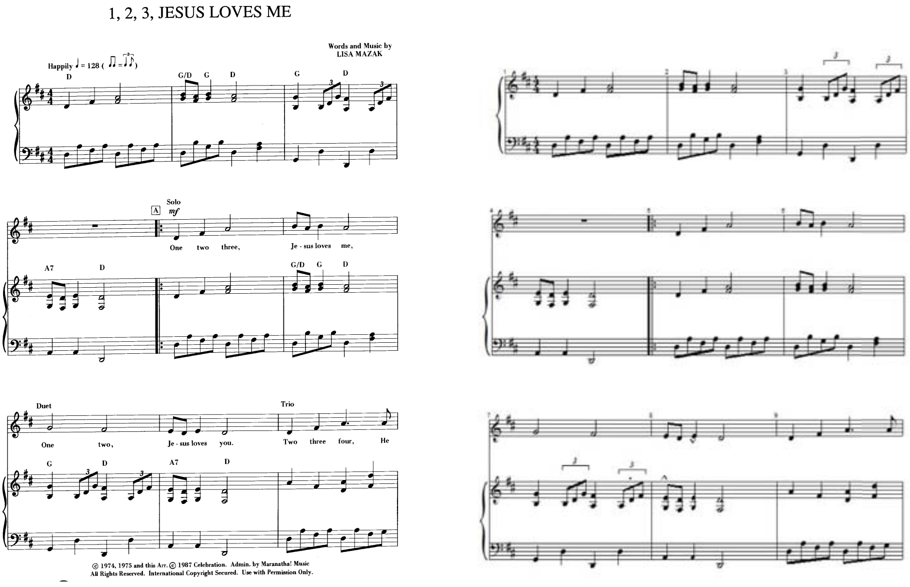


Fig. 4. Example of music notation recognition in the Braille Score computer program. Original score is given on the left, recognition result is on the right.

granular computing and a granular representation of information, cf. [1,5]. Music notation recognition includes wide range of information processing: starting with low level data (represented in a form of raster image acquired from peripherals), through data segmentation, feature extraction and symbols classification, till mining elements of information structure with an emphasis given to building a structure. The structure of information is a fundament of further music information processing like correction of recognition errors, editing, selection, copy/paste, find/replace, conversion to different formats, transposition, etc.

An example of music notation, recognized by the Braille Score computer program[2], is given in Figure 4, cf. [3]. Several elements of an original score are not recognized: title and composer, global rhythmic data, guitar chords, lyric, footnote. Some recognized elements are displayed differently then in original score, though correctly, e.g. triplets. It should be underlined that recognition of a score consists of locating and finding symbols (notes, rests, clefs, chromatic symbols, bar lines, staff lines, etc.) and uncovering relations between symbols. Uncovered relations allow for building a score structure (systems, measures, voices, etc.).

2.2 Selection in Music Notation

Selection in music notation is a fundamental operation based on a structure of information. It could be interpreted as a kind of selection in a vector

[2] The Braille Score computer program was developed in frames of State Committee for Scientific Research Grant realized at the Warsaw University of Technology.

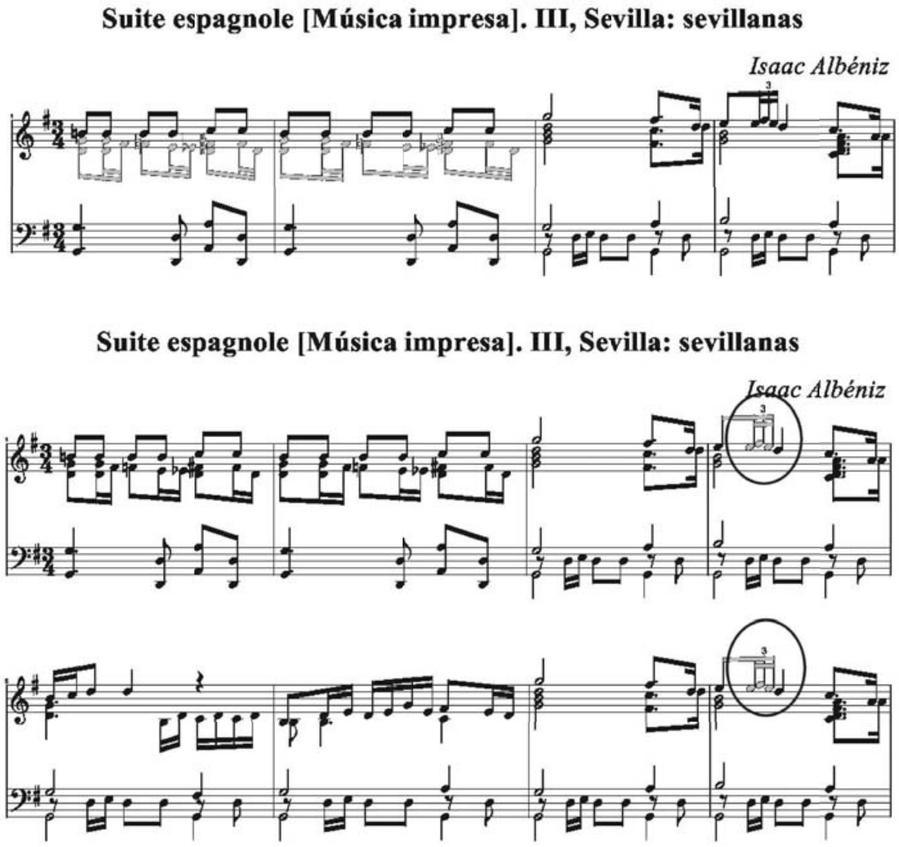

Suite espagnole [Música impresa]. III, Sevilla: sevillanas
Isaac Albéniz
3
Suite espagnole [Música impresa]. III, Sevilla: sevillanas
Isaac Albéniz
3
3

Fig. 5. Selection of the lower voice in two measures (upper part of the image) and triplets (lower part of the image)

graphics. However, since it is basic operation for further structural operations (e.g. Copy/Paste, Find/Replace, Transpose, etc.), it must engage domain knowledge. Group of symbols without relations between them make little sense in music notation. Therefore, a selection of a group of symbols without a structure they create would be useless as a basis for further processing.

Examples of selection in a music notation are given in Figure 5. The selected part of the lower voice is displayed in gray in upper part of Figure while selection of triplets in the upper voice of the score is shown in the lower part of the Figure. In both cases selection is deeply hooked on the structure of the score.

2.3 Transposition in Music Notation

Transposition is done either on a whole score or on given, selected, part of a score. It moves transposed notes to new positions preserving music interval between

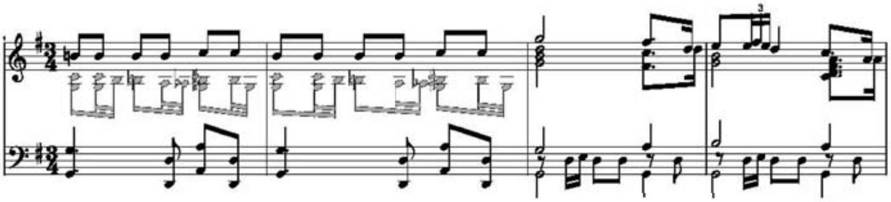


Fig. 6. Transposition of a section of lower voice

source and target. However, geometrical features may be preserved, e.g. they may change stem direction, relative vertical placement, drop or get chromatic symbols, etc.

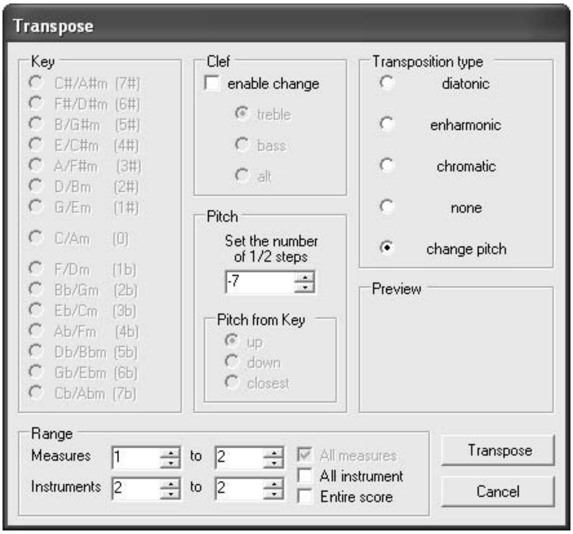
An example of transposition of the part of voice is shown in Figure 6. Selection of this part of lower voice is shown in Figure 5. The selected part of the voice is transposed 7 halftones down. Note that interval of vertical shift in terms of lines/spaces is not the same for every transposed note. This geometrical irregularity of the operation is caused by a nature of processed data structure.

3 Communication

Man-machine communication is aimed to control data processing. Simple operations done on processed data, as shown in section 1, do not need a sophisticated user interface. Simple mouse and keyboard operations as, for instance, mouse click and drag, drag and drop, navigation keys with Shift and Ctrl keys of keyboard allow for full control of low level data processing.

However, when data create a complex information structure, common simple tools are far inadequate for communication. For instance, transposition discussed in section 2.3 requires a record of domain information not definable with simple hardware or software tools (mouse, keyboard, standard menus or toolbars). In some cases standard keyboard/mouse operation allow for reasonable selection in a score, e.g. selection of a measure on a stave. Yet, more specific selection requires more complex environment. For instance, a section of a voice could be selected with standard mouse click on the beginning note and Shift mouse click on the ending note of the section. This operation needs a special *voice mode* in order to select a voice section and not, for example, a section of a whole stave.

A special dialog box is shown in Figure 7. It was designed for Braille Score computer program to define the operation of transposition. It allows for both selection of a part of a score and transposition of this part. Selection and transposition shown in Figures 5 and 7 define the same transposition as shown in Figure 6. Man-machine communication discussed here depends deeply on understanding transposition features and factors described in the dialog box. It also depends on ability of embedding them in the structure of music information as well, cf. [1,2,6].



Transpose

Key
○ C#/A#m (7#)
○ F#/D#m (6#)
○ B/G#m (5#)
○ E/C#m (4#)
○ A/F#m (3#)
○ D/Bm (2#)
○ G/Em (1#)
○ C/Am (0)
○ F/Dm (1b)
○ Bb/Gm (2b)
○ Eb/Cm (3b)
○ Ab/Fm (4b)
○ Db/Bbm (5b)
○ Gb/Ebm (6b)
○ Cb/Abm (7b)

Clef
□ enable change
● treble
○ bass
○ alt

Pitch
Set the number of 1/2 steps: -7

Pitch from Key
● up
○ down
○ closest

Transposition type
○ diatonic
○ enharmonic
○ chromatic
○ none
● change pitch

Preview

Range
Measures 1 to 2 ☑ All measures
Instruments 2 to 2 □ All instrument
□ Entire score

Fig. 7. Transposition dialog box in Braille Score

References

1. Homenda, W.: Towards automation of data understanding: integration of syntax and semantics into granular structures of data. In: Fourth International Conference on Modeling Decisions for Artificial Intelligence, MDAI 2007, Kitakyushu, Japan (2007)
2. Homenda, W.: Integrated syntactic and semantic data structuring as an abstraction of intelligent man-machine communication. In: ICAART - International Conference on Agents and Artificial Intelligence, Porto, Portugal, pp. 324–330 (2009)
3. Luckner, M., Homenda, W.: Braille Score. In: Chen, Y., Abraham, A. (eds.) Sixth International Conference on Intelligent Systems Design and Applications, ISDA 2006, pp. 775–780. IEEE, Los Alamos (2006)
4. Macukow, B., Homenda, W.: Methods of Artificial Intelligence in Blind People Education. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M., et al. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 1179–1188. Springer, Heidelberg (2006)
5. Pedrycz, W., Skowron, A., Kreinovich, V.: Handbook of Granular Computing. J. Wiley, New Jersey (2008)
6. Tadeusiewicz, R., Ogiela, M., Szczepaniak, P.: Notes on a Linguistic Description as the Basis for Automatic Image Understanding. Int. J. Appl. Math. Comput. Sci. 19(1), 143–150 (2009)

Memory Usage Reduction in Hough Transform Based Music Tunes Recognition Systems*

Maciej Hrebień and Józef Korbicz

Institute of Control and Computation Engineering
University of Zielona Góra, ul. Podgórna 50, 65-246 Zielona Góra
{m.hrebien,j.korbicz}@issi.uz.zgora.pl
http://www.issi.uz.zgora.pl

Abstract. This work presents a proposition of the Hough transform's accumulator *on-line* compression method in the context of music tunes identification system. One can also find here a short discussion on the method's properties as well as experimental results obtained during the research.

1 Introduction

The growing industry of multimedia devices is the fact that is hard to miss nowadays. The cell-telephonization of our community is also very noticeable and currently it is tremendously hard to find a person that is not using cell phone in daily living. The development of both areas as well as the increase of abilities and computational power of hardware causes the development of new cell telephony gadgets as well hardware (like MP3 decoders) as software (like video games). Among the mentioned gadgets, systems that try to support a telephony client in automated music tunes recognition based on a few second sample recorded by cell telephony audio subsystem are starting to appear [2,4,10,12]. The systems, regardless of their easiness of use by the end-user (dial a dedicated number and put your cell near sound source), are in many cases based on advanced signal processing and recognition techniques which are supported by a large database of tunes. One of the propositions in solving the problem of automated tunes recognition is the technique presented in [6,7] which is based on the Hough transform [1,11] adapted for localization and matching of a given set of sound features using the transform's distinctive accumulator's voting technique (see Fig. 1 for instance). The mentioned accumulator of the transform and its construction is one of the key issues that have to be considered when setting up the system for work. The increase of matching precision forces the right management strategy of the information collected in the accumulator's cells to be applied to make the system work more compact in the sense of memory resources indeed.

* This work has been supported by the Ministry of Science and Higher Education of the Republic of Poland under the project no. N N519 4065 34 and the decision no. 9001/B/T02/2008/34.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 484–491, 2010.
© Springer-Verlag Berlin Heidelberg 2010

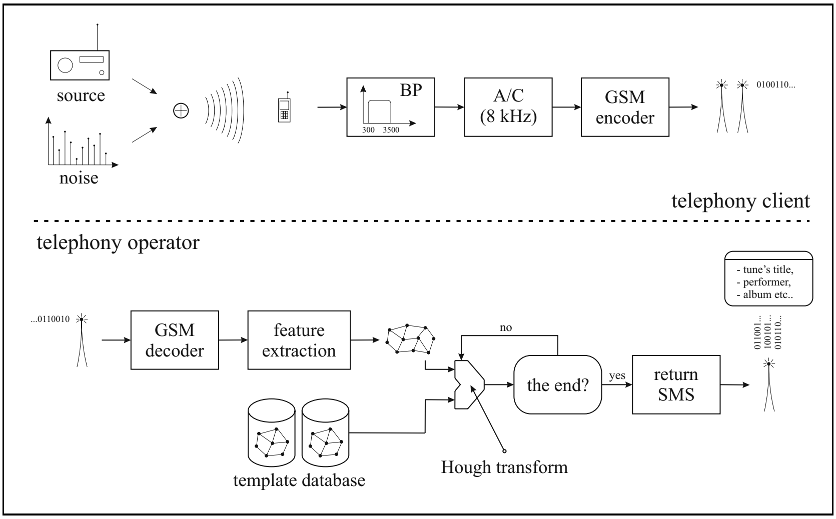


Fig. 1. Schematic diagram of the considered music tunes identification system

In this work a method of the Hough transform's accumulator *on-line* compression algorithm in the context of cell telephony based music tunes recognition system is presented. One can also find here experimental results showing the efficacy of the proposed solution as well as a short discussion on the method's advantages and disadvantages.

2 Motivation

The main motivation to consider the problem of the Hough transform's accumulator compression are practical remarks. For instance, the music identification problem considered in this work needs about 50 MB of operating memory for a typical $3 \div 5$ minute music tune to construct the accumulator with 20 ms and 7.8125 Hz cell granularity. As far as this portion of needed memory seems to be not much in todays Giga-Byte and relatively not very expensive memory chips, in case of a large number of requests at the same time, very long tunes (what can be for example observed in classical music where 20 to even 30 minute compositions are nothing extraordinary), hardware based solutions that are small sized and/or have very small memory resources or simply a higher resolution is needed, a problem of too low system memory resources can occur very often. Thus, it is worth to consider a dedicated Hough transform's accumulator compression method that will reduce the memory demands using the specifics of the data collected in its cells. In case of lossy compression method proposed and presented here it is of course a desired feature to have a precisely reconstructed accumulator's content with a possibility to steer the level of precision as well as the compression factor and speed of the operation. It is also desired to preserve the speed of the voting process as much as it is possible and also to give the opportunity to parallelize some of the operations of the method to effectively make a use of multi-core processing units and/or multi-threaded operating systems. Worth noticing is also the aspect of local maxima localization at the end of the voting process that could be simplified to a smaller area of search in opposition to the whole accumulator look through known from the classical solution [1]. And the last thing is that the method must work in the *on-line* fashion, that is it starts with a small memory allowing the voting process to be performed just like in the standard Hough transform, thus a quite simple approach compromising the compression efficacy and the speed of operation is desired.

As far as some techniques can be found in literature with the fast Hough transform being the closes relative to the considered problem [3,8,9], its way of operation does not meet all of the mentioned assumptions. Moreover, it uses more advanced data structures, namely trees, which are not easy do implement in the pointer-less programming environments and can perform quite poor in such implementations, it needs to propagate votes up the tree (from root to leaves) which is also not very effective, it introduces a proper voting sub-space split threshold selection problem, which can cause uncontrolled tree overgrowth and what can be a problem in low memory resources systems, and it needs the voting process to be repeated if tree undergrowth occurs or non-one-pass implementation is used. Thus, the above defined requirements and the lack, according to the authors current knowledge, of satisfying solution in literature are the base to construct the *on-line* Hough transform's accumulator compression method, which will be presented in detail in the next section.

3 Proposed Solution

The construction of the compression method is based on the notice that the information collected in the accumulator has peak-like character creating local maxima (smaller or larger in the sense of height as well as width), which values cumulated in the cells define the level of similarity between the feature function or feature set and the searched object. Thus, the presented here compression algorithm can use this fact extensively to model the content of the Hough transform's accumulator more or less precisely depending on demands. It is also good to notice here on a margin that the mentioned peak-like characteristic of the collected information can be observed in all the versions of the Hough transform due to its design and means of operation. Thus, the proposed here compression method can be used after adaptation in other approaches, but will be presented here in the context of music tunes identification problem considered in this work. Analyzing closely the peaks that can be observed in the accumulator's cells it can be noticed that the following function:

$$f(n) = \frac{1}{|n| + 1}, \; \textit{where } n \textit{ is any integer number}, \tag{1}$$

can be a relatively good and simple approximator, the base function for the compression mechanism that do not require a significant computational power.

```
vote($position$, $value$):
  – $l$ = decide which line the vote is addressed to based on the $position$,
  – wait and lock the $l$ line with a semaphore,
  – prepare a thread to execute the following operations:
      • if the buffer for the $l$ line is full, update the structures for the $l$ line with the
        $\theta$ threshold flushing the buffer,
      • insert the defined by $position$ and $value$ vote to the $l$ line's buffer,
      • release the semaphore unlocking the $l$ line,

update($l$, $\theta$):
  – $t$ = create a temporary line filled with zeros,
  – for each $p$ structure defining the $l$ line:
      • add the $f(n) = \frac{1}{|n|+1}$ function at the $t$ line's $p_{pos}$ position scaling it by $p_{val}$
        value (height of the peak) with $n$ equal to $0, \pm 1, \ldots, \pm(p_n - 1)$ (width of the
        peak) to the values collected already in the $t$ line; if two peaks meet, use the
        MAX function to decide which values have to be preserved,
  – remove all the $p$ structures defining the $l$ line,
  – for each $v$ vote from the $l$ line's buffer:
      • add the $v_{val}$ valued vote at the $v_{pos}$ position to the $t$ line,
  – estimate the level of information in the $t$ line using: $\mu = \sqrt{\frac{1}{N}\sum_{i=1}^{N} t(i)^2}$, where
    $N$ is the line length,
  – for each $t$ line's peak (local maximum) that is crossing the $\mu 10^{\theta/20}$ threshold:
      • estimate the $n$ value of the $\frac{1}{|n|+1}$ base function analyzing the left and right
        side of the peak until a greater value then the foregoing is encountered,
      • store a new structure filled with the peak's position, height (accumulator's
        cell value) and width ($n$) in the line's defining structures area,
  – reset the "next free element" index of the $l$ line's buffer,
  – remove the $t$ temporary line from memory,

global_maximum($A$):
  – update each line defining the $A$ accumulator (flush the voting buffers),
  – find the maximal value among the structures defining each of the line,
```

Fig. 2. Hough transform's accumulator *on-line* compression algorithm

Due to the fact that the Hough transform's accumulator is in many cases multidimensional (two-dimensional in the considered problem where shift in time and frequency has to be detected), it must be split into a set of row vectors (lines) to simplify the algorithm and make it independent of the number of dimensions. The split operation is recommended to be performed along the smaller dimension to reduce the number of vectors and to increase the compression factor. Since the peaks that cross an arbitrary given detection threshold will be converted into (position, height, width) structure that will replace the peak and since the line the peak was detected in will not have a physical representation, to equip the algorithm with a fast voting mechanism a small voting buffer has to be

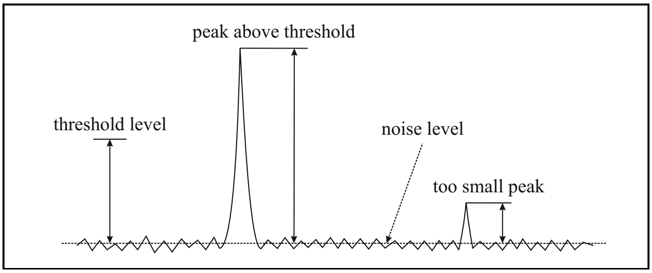


Fig. 3. Peak detection illustration in an exemplary accumulator's line

attached to each of the lines. Thus, after preparation of structures that will define the content of a line, and voting buffers to partially preserve the speed of the transform, the voting process can be started, which decides in each iteration which line a vote is addressed to based on the cell coordinates and inserts the vote to a next free position in the buffer that is attached to the line. When the buffer is already full, the structures that define the peak-like content of a line have to be updated to make a place for subsequent votes. In case of the first time update a temporary line filled with zeros has to be created, and in case of subsequent updates its content has to be additionally reconstructed using the line defining structures to which the data collected in the buffer are then added preserving the voting order. After the temporary reconstruction of a accumulator's row vector, all the peaks above the noise level in the line that cross a given threshold level (see Fig. 3 for illustration) are converted into short (position, height, width) structure representation. The temporary line can be then released or unlocked if this portion of memory is a shared resource and the voting buffer can be reseted to allow subsequent votes cumulation (see the algorithm in Fig. 2). It is good to notice here that the line update stage can be evaluated in a separate thread outside the main voting loop (in its background) to make a better use of multi-core hardware and/or multi-threaded operating systems, certainly with a proper synchronization mechanism involved to lock the line that is in the update state. The proposed and presented here the buffer-like means of votes series collection tries to preserve the speed of the voting process, but the size of the buffers as well as the peak detection threshold is a compromise between the mentioned speed of the method, its compression factor (in the sense of saved memory) and quality of the information collected in the accumulator (see exemplary results in the next section). It is also good to note here that the structures defining the content of a line contain the height of a peak information, which is nothing else than the accumulator's local maximum. Thus, after the voting process and the last buffer flushing update the structures can be used to find all the local and the global maximum without entire accumulator look through, what increases the speed of this final stage in comparison to the classical approach [5].

4 Experimental Results

The conducted experiments were about to measure the influence of the proposed compression mechanism on the overall efficacy of the Hough transform based matching algorithm. The variables that were observed include matching efficacy ratio, response time and peak-signal-to-noise ratio defined as:

$$PSNR = 20 \log_{10} \left(\frac{MAX(A)}{RMS} \right), \tag{2}$$

where RMS is the root-mean-square (quadratic mean of the data collected in the accumulator) defined as:

$$RMS = \sqrt{\frac{1}{TF} \sum_{t=1}^{T} \sum_{f=1}^{F} A(t,f)^2}, \tag{3}$$

and where T and F define the size of the A accumulator as well in time as in frequency dimension.

The experiments were performed on 1000-element database of different kind of music genre with 100 5-second length randomly chosen samples to identify. The samples were disturbed by 0dB white noise and the time trials were performed using sequential execution (without multi-threading) repeated 10 times per trial for a better average on three different CPUs. The peak detection threshold was ranging from 10dB to 30dB with 5dB steps and the buffer size was ranging from 128 to 2048 elements with subsequent power of 2 steps. The implementation was written using the C programming language and the tests were performed at operating system's user level, that is with other system tasks working in background, and under control of software written cell telephony simulator (thus no cell telephony hardware was used). The obtained results are given in Tab. 1.

As it can be observed in the obtained results, for an average $3 \div 5$ minute music tune the method needs about 50 MB of memory for the accumulator with 7.8125 Hz and 20 milliseconds cell granularity without any compression mechanism. However, the memory demands can be reduced if needed from about 1.4 to nearly 26 times depending on settings using the proposed *on-line* accumulator's content compression algorithm at a cost of buffers' size depending speed decrease and data reconstruction accuracy. It it good to notice here that over 10 times compression ratio can be achieved in the considered problem using the settings that well preserve the accumulated information, that is 256-element buffer and 15dB peak detection threshold, at the cost of $2 \div 4$ times response time decrease depending on the used CPU and memory chips latencies. The obtained results also illustrate how the compression mechanism affects the sharpness of the accumulator's content revealing the additional noise reduction property of the proposed algorithm, which increases the accumulator's PSNR ratio by a value that highly depends on the used settings and ranges from about 3 up to 27 dB in the obtained experimental results. Certainly, the extreme cases of both the compression as well as the PSNR ratio, not always have a positive effect

Table 1. Experimental results for SNR = 0dB (upper) and corresponding time trials for Intel Pentium III 500 MHz / Intel Pentium D 2.8 GHz / AMD Athlon64 3000+ 1.8 GHz microprocessors (lower)

average efficacy [%], PSNR [dB] and peak memory usage [kB]

	peak detection threshold				
buffer	10 dB	15 dB	20 dB	25 dB	30 dB
128	93 / 49 / 4268	93 / 50 / 2756	90 / 53 / 2267	85 / 55 / 2210	84 / 59 / 1946
256	98 / 49 / 6010	98 / 51 / 4534	95 / 53 / 4478	92 / 56 / 4254	92 / 61 / 3481
512	98 / 49 / 9491	97 / 51 / 8965	98 / 54 / 8855	96 / 57 / 8040	93 / 63 / 6142
1024	97 / 49 /17896	98 / 51 /17876	98 / 54 /17362	97 / 59 /14896	97 / 65 /10644
2048	98 / 49 /35748	98 / 51 /35695	98 / 55 /33689	98 / 61 /27276	96 / 66 /19185

without compression = 98% / 46.12 dB / 50363 kB

average computation time [s]

	peak detection threshold				
buffer	10 dB	15 dB	20 dB	25 dB	30 dB
128	13.7 / 4.5 / 3.5	11.4 / 3.5 / 2.9	9.9 / 2.9 / 2.5	9.0 / 2.5 / 2.2	8.5 / 2.2 / 2.1
256	7.5 / 2.3 / 1.8	6.3 / 1.8 / 1.5	5.6 / 1.5 / 1.3	5.2 / 1.3 / 1.2	5.1 / 1.3 / 1.2
512	4.5 / 1.3 / 1.1	3.9 / 1.0 / 0.9	3.6 / 0.9 / 0.8	3.5 / 0.8 / 0.8	3.4 / 0.8 / 0.8
1024	3.2 / 0.8 / 0.7	2.9 / 0.7 / 0.7	2.7 / 0.6 / 0.6	2.7 / 0.6 / 0.6	2.6 / 0.6 / 0.6
2048	2.6 / 0.6 / 0.6	2.4 / 0.5 / 0.5	2.3 / 0.5 / 0.5	2.3 / 0.5 / 0.5	2.3 / 0.5 / 0.5

without compression = 2.9798 s / 0.4263 s / 0.4277 s

on the system's matching efficacy and response time but are mentioned here to show the method's experimentally determined possibilities.

5 Conclusions

The proposed and presented in this paper method of the Hough transform's accumulator compression is a simple and effective way of memory reduction in the *on-line* fashion. The buffering mechanism and the fact that some of the operations can be parallelized preserves the speed of the voting process on a tolerable level, and the organization of structures that define the content of a line allows a fast local maxima localization practically in any moment of the being executed algorithm. An interesting feature of the proposed algorithm is also the fact that the method reduces the noise that is present in the accumulator's cells. It happens because all the peaks that do not cross the peak detection threshold during the line update stage are omitted and threated as noise, what increases the sharpness of the information stored in the accumulator enhancing the most significant maxima. Unfortunately, beside the mentioned advantages the method has two drawbacks. Firstly, the line update stage is preceded with a temporary decompression and line reconstruction after which the buffered votes are moved to the line and the main part of the compression starts, what lengthens in time the voting process in comparison to the standard version of the transform, and too small buffers can additionally intensify the delays. Secondly, the proposed algorithm is a lossy compression method, what means that if the combination of peak detection threshold and the size of the buffers is chosen to aggressively compress the data collected in the accumulator, losses and information leakage can occur (from the least to the most significant), what will have effect in efficacy decrease of a solution the compression mechanism is used in. Nevertheless, a reasonable choice of the mode (size of buffers / threshold level combination) the method can work in allows to reduce the memory demands and preserve the quality of the information stored in the accumulator on an acceptable level with a not very time consuming manner. Thus, the method, in the authors' opinion, can be used with success in all the implementations where the problem of low memory resources occurs, the time of operation is important (but has lower priority over the memory), the voting process should be relatively fast and single (without repetitions) and the information stored in the accumulator preserved at the same time on an acceptable level. It is also good to underline at the end the fact, that the method can be used after adaptation in other problems and is not limited to the music tunes identification issue considered in this work.

References

1. Ballard, D.: Generalizing the Hough transform to detect arbitrary shapes. Pattern Recognition 13(2), 111–122 (1981)
2. Dannenberg, R., Hu, N.: Pattern discovery techniques for music audio. In: Proc. of the 3rd Int. Conf. on Music Information Retrieval, Paris (2002)
3. Guil, N., Villalba, J., Zapata, E.: A fast Hough transform for segment detection. IEEE Trans. on Image Processing 4(11), 1541–1548 (1995)
4. Haitsma, J., Kalker, A.: A highly robust audio fingerprinting system. In: Proc. of the 3rd Int. Conf. on Music Information Retrieval, Paris (2002)
5. Hough, P.: Method and means for recognizing complex patterns, US Patent no. 3,069,654 (1962)
6. Hrebien, M., Korbicz, J.: Hough transform in music tunes recognition systems. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 588–596. Springer, Heidelberg (2008)
7. Hrebien, M., Korbicz, J.: Hough and fuzzy Hough transform in music tunes recognition systems. In: Proc. of the Int. Conf. on Adaptive and Natural Computing Algorithms. LNCS (LNAI), pp. 479–488. Springer, Heidelberg (2009)
8. Jeng, S., Tsai, W.: Fast generalized Hough transform. Pattern Recognition Letters 11(11), 725–733 (1990)
9. Li, H., Lavin, M., LeMaster, R.: Fast Hough transform. In: Proceedings of the 3rd Workshop on Computer Vision: Representation and Control, pp. 75–83 (1985)
10. Prado, B.: Finding structure in audio for music information retrieval. IEEE Signal Processing Magazine 23(3), 126–132 (2006)
11. Smereka, M., Duleba, I.: Circular object detection using a modified Hough transform. Int. J. of Appl. Math. and Comp. Science 18(1), 85–91 (2008)
12. Weinstein, E., Moreno, P.: Music identification with weighted finite-state transducers. In: Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing, vol. 2, pp. 689–692 (2007)

CogBox - Combined Artificial Intelligence Methodologies to Achieve a Semi-realistic Agent in Serious Games

David Irvine and Mario A. Gongora

Institute of Creative Technologies, De Montfort University, The Gateway, Leicester,
LE1 9BH, UK
dirvine@dmu.ac.uk, mgongora@dmu.ac.uk

Abstract. CogBox is a system created to demonstrate a theory on Artificially Intelligent Agents. This paper describes this theory, explaining how it improves upon existing AI Agents. We draw upon already well established AI concepts, such as Fuzzy Logic and the Belief, Desire, Intention model, as well as Finite State Machines and discuss their limitations. We present a novel combination of these techniques to overcome these limitations.

Keywords: BDI, Fuzzy Logic, Finite State Machines, Fuzzy Agents, Agent Behaviour, Hybrid Systems.

1 Introduction

As AI Agents progress forward, they continue to be controlled almost exclusively by one AI methodology in any given implementation, be it Belief Desire Intention [1], Fuzzy Logic [2], or Neural Networks [3]. While these methods function, applying them to an entire Agent often requires the method to be substantially modified, creating a very complex implementation. We believe that a hybrid system of several AI systems will result in a simpler implementation and an Agent that behaves in a semi-realistic manner.

In this paper we discuss how these different systems have limitations in the field of behaviour simulation, and also explain how our hybrid system allows some AI methods to complement each other. This paper is organised as follows; Section 2 provides a background of the Core Techniques that we intend to combine, Section 3 details our proposed hybrid method, in Section 4 we analyse our method, concluding in Section 5 and noting any further work in Section 6.

2 Core Techniques

A number of techniques already exist within the field of AI that allow for the simulation of semi-realistic Agents. As they stand on their own, existing techniques often contain inherent limitations that make it complicated to achieve a fully featured behaviour simulation. It is our aim to combine these techniques, in order to create a more efficient system. We have concentrated on the following existing techniques.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 492–499, 2010.

© Springer-Verlag Berlin Heidelberg 2010

2.1 Belief Desire Intention

Belief/Desire/Intention (or BDI) is a method of modelling behaviour in rational agents. At its core, it is seen as giving the agent “mental attitudes of Belief, Desire and Intention, representing, respectively, the information, motivational, and deliberative states of the agent.” [4] The BDI model is based on work in the areas of both Philosophy and Cognitive Science, including work on Rational Agents. [5] Rational agents have bounded (i.e. limited) resources and usually incomplete knowledge about their environment. This incomplete knowledge is usually defined as the agents’ belief network.

BDI has been implemented in several research models, as well as applied in several situations such as Air Traffic management [4], military combat simulation [6], and has subsequently been implemented into frameworks such as the Procedural Reasoning System [7], Jadex [8] and JACK Intelligent Agents [9].

2.2 Finite State Machines

Finite State Machines have become more frequently used in today's computer games (Age of Empires, Half Life) as a method for generating AI behaviour. In most cases, NPC behaviour can be modelled in terms of mental states, and hence, FSMs are a natural choice for game developers when designing NPCs [10]. The programmer predefines each and every state. When the Agent enters this state it follows the rules laid down by the programmer. The Agent can then enter another state and follow another set of rules, or stay in the same state repeating rules infinitely.

If well thought out, this approach can lead to convincing AI that can carry out tasks with some degree of success, however, as the amount of states increases, so does the complexity of the FSM. Code complexity in a FSM increases significantly as the system scales up with more behaviours [11], and iterating through all the state options may become too computationally expensive for a system required to run in real-time. In our theory, we aim to use FSMs to deal with very specific states that we consider necessary to be controlled directly to avoid behaviours that would be inappropriate.

3 Hybrid Method

The Core Techniques reviewed are all capable of producing semi-realistic Agents, but it is often the case that an excessive amount of adjustment of these Techniques is necessary to achieve all required aspects of behaviour. By combining different technologies we are capable of creating semi-realistic agents with purer implementations of each individual technology. [11] noted that research should be undertaken to determine if hybrid models of Finite State Machines performance and a BDI Agents’ ease of design and coding, and we aim to address this.

Our Agents are primarily controlled by a BDI system; this is our emulation of human reasoning. The virtual world in which our Agents exist is simulated in a crisp fashion, therefore it is necessary to simulate the belief system of our Agents, adding uncertainty and imperfection to their beliefs. We pass these beliefs on to the BDI system to enable our Agents to have a more realistic view of the virtual world. A FSM is given overriding control of the BDI system for situations where direct action must be taken with no deliberation, providing our emulation of human instincts.

3.1 Belief Simulation

It is important to note that an autonomous agent's decisions and plans are influenced by the information it receives about the world in which it is operating [6]. For the purposes of attempting to semi-accurately simulate real-world behaviour, a purely computational agent is at an immediate advantage; without the additional stage of filtering data that the agent receives from the Virtual Environment, the agent is automatically given a cognition of a situation that no real-world creature is capable of obtaining – Agents automatically “see and understand everything”, and as such we must inhibit this ability. Work on simulating belief systems of agents has already been undertaken by [6]. They suggest “conceptualizing the agent’s activities in terms of three main steps: detection, measurement and interpretation”. We carry over this idea to a belief simulation implementation in CogBox. As the Agent picks up (detects) new information it should be measured in such a way that we can determine distance from the Agent; the level of accuracy will be inversely proportional to the distance.

Once a level of accuracy has been determined we are able to affect the quality of information that the agent receives. This can be defined as the “interpretation” stage. Given a set of imperfect data the agent must make a decision on how to interpret it. If an object is visible in the distance to the Agent, it is less likely to interpret it correctly as opposed to a nearby object. By determining how imperfect data it has received is, the Agent is capable of making a decision of what to do with the data. This decision making process is handled by our BDI implementation.

3.2 Planning and Decision Making Using BDI

Decision making via a BDI system enables us to control our Agent’s higher level functions, such as social interaction, complex problem solving, and other behaviours that are not otherwise covered by human instinct.

The basic interpreter cycle for a BDI agent acts thus; The agent updates its beliefs, and then updates its available options. Once it has done this it examines its desires, matches them against these options and proceeds to deliberate over them, before updating its intentions and executing the plans to facilitate those intentions. In the final steps, the interpreter removes successful and impossible goals and intentions. By expanding on the abstract interpreter suggested in [4], we are able to form a method that allows our Agents to make decisions about the virtual world based on imperfect values it receives, and then form plans to determine its actions in this virtual world. Methods to expand the abstract interpreter already exist in several implementations, including JADEX [8], METATEM [12] and AIL [13]. These existing implementations provide a software framework for the creation of goal-oriented agents [8], and as such could be used to implement BDI in our combined system.

We can avoid the issue of a BDI Agent ignoring what would seemingly be a high-priority behaviour in favour of another entirely by extracting behaviours from the BDI Agent that humans would consider “priority behaviours”, or, “instinctual behaviours”.

3.3 Instinct Emulation Using FSMs

Instinct is often an overriding factor in human behaviour. In situations that are life threatening, instinct comes into play and determines a behaviour to remedy the situation with little reasoning or thought for consequences. While instincts can be controlled, it can be more common (and indeed reasonable) to let an instinct take hold and dictate your actions.

Finite State Machines are capable of making an Agent act a certain way in a finite amount of states with little processing requirement, therefore making them very fast when the list of states is small. The larger a FSM becomes, the less manageable it is for the systems programmer, and the more states the Agent must iterate through to determine its behaviour. For this reason a FSM should only be responsible for the states that we define as being “instinctual”. Our method contains a small set of basic instincts, dealing only with very specific cases. After the Agent receives data from the belief simulation, this is passed to the BDI system for evaluation. Once evaluated, the FSM is able to judge the state that the Agent is currently in. If the Agents’ state is recognised as “instinctual”, the FSM seizes control from the BDI. Once the Agent’s state has changed, the belief simulation system generates a new set of beliefs.

4 Analysis of Hybrid Method

When analysing our hybrid BDI-FSM system we see that combining AI methods produces a technique wherein methods that are better suited to simulating behaviour in certain situations, such as Finite State Machines simulating instincts, can override other methods when they are not as well suited. We can now define a test scenario in which to place a group of Agents. The behaviours of these Agents can then be modelled and presented in order for us to view any possible grouping of technologies that is preferential in terms of complexity.

4.1 Test Scenario

For this analysis, a number of Agents will be placed at random in a virtual environment; and each of these Agents have a basic need for social interaction. In addition to this basic need, they follow the instinctual desire to avoid outbreaks of fire. Each Agent is given 4 traits (Wealth, Education, Authority and Fitness), which will each be assigned a value between 0 and 1.

By varying the level of each trait that every Agent has, we are able to create Agents that have different degrees of compatibility with each other. When created, each of the Agents is free to move about within the virtual environment at a certain speed. As the Agents move they look in front of them with a limited visual radius and focal length. Any other Agents that they encounter within this visual range they should approach and “interact” with. This interaction allows for a calculation of how compatible each Agent is with each other. This is a simple calculation based on comparing the degree of membership of each trait each Agent has:

$$\text{Compatibility} = \{\ 1 \text{ if } \delta\text{Traits} \le 1\ ,\ 0 \text{ if } \delta\text{Traits} > 1\ \}$$

Where:

$$\begin{aligned}
\delta\text{Wealth} &= \text{Agent}_1.\text{Wealth} - \text{Agent}_2.\text{Wealth} \\
\delta\text{Education} &= \text{Agent}_1.\text{Education} - \text{Agent}_2.\text{Education} \\
\delta\text{Authority} &= \text{Agent}_1.\text{Authority} - \text{Agent}_2.\text{Authority} \\
\delta\text{Fitness} &= \text{Agent}_1.\text{Fitness} - \text{Agent}_2.\text{Fitness} \\
\delta\text{Traits} &= \delta\text{Wealth}+\delta\text{Education}+\delta\text{Authority}+\delta\text{Fitness}
\end{aligned} \tag{1}$$

If the other Agent is “compatible”, this Agent remains stationary as it has completed its task of finding a compatible Agent. If the other Agent is not compatible, this Agent moves off in search of othes. At randomly spaced intervals a fire breaks out at a random location in the virtual world. Any Agent near to the fire should move away from it; after which they return to their default searching behaviour.

4.2 Modelling of Test Scenario

We can model the different behaviour systems of the Agents to validate our proposed hybrid method to reduce system complexity and increase realism.

4.2.1 FSM Based Model

This model demonstrates all the possible states that our Agent can enter in our test scenario, as well as determining the transitions that can occur to move between states. In every state it is possible to transition to the “Escaping Fire” state, and it should be noted that the addition of states would cause the model to become increasingly complex, as those states would also need to transition to “Escaping Fire”. Also note we may not have included all the possible states that should exist; without repeated tests of the scenario, not all possibilities may have presented themselves.

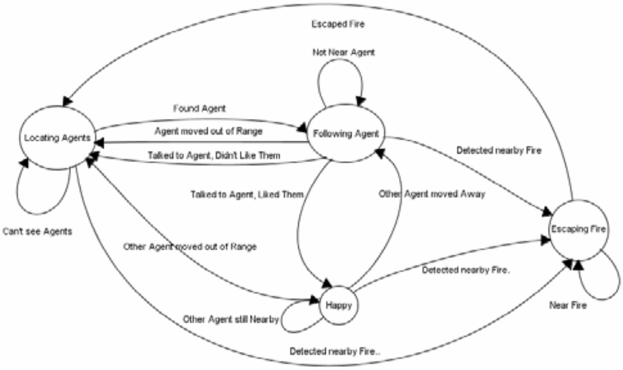


Fig. 1. FSM Model of Test Scenario

4.2.2 BDI Based Model

To implement our test scenario using BDI we need the following components:

Beliefs: { Location of Agents, Compatibility with other Agents, Location of Fire }
Desires: { Socialise with others, Avoid Fire }
Intentions: { Move to Agents, Talk to/determine compatibility with other Agents, Move away from Fire }

This model does not deal with the complexities of adding values of "Commitment" to each Intention. It should also be noted that the Intentions listed here are conceptual in basis, as a BDI Agent only has one Intention at any time, selected from its Desires. We also do not deal with the priority of any Desire; without this an Agent can follow a Desire blindly when another should clearly take precedent. While this model is not complex, it highlights one limitation of BDI; namely that the computational cost of each BDI Agent is significantly higher than for a FSM implementation [11].

4.2.3 Proposed Hybrid Model

Our hybrid model combines the most appropriate components from FSM and BDI to reduce the complexity of the system and the computational cost of running the test.

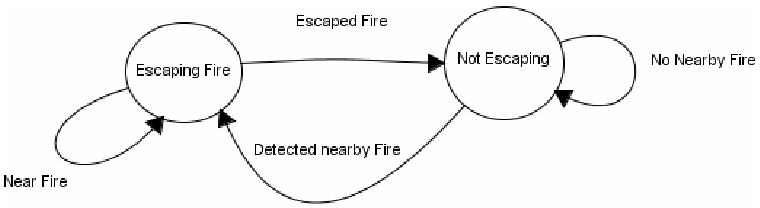


Fig. 2. FSM Model as part of Hybrid solution

This FSM component deals with situations where we believe an "instinctual behaviour" is necessary to resolve the situation. In this case the need for an Agent to avoid nearby fires has been defined in this category.

In this model, the BDI system handles any other situations when the FSM is in the "Not Escaping" state:

Beliefs: { Location of Agents, Compatibility with other Agents }
Desires: { Socialise with other Agents }
Intentions: { Move to Agents, Talk to/determine compatibility with other Agents }

4.3 Analysis of Hybrid Model Effectiveness

We have observed that using FSM can result in a system that quickly becomes complex to implement and manage. Every state must be pre considered, as well as every possible transition between states. As noted, any additional states increases the complexity of the model significantly, requiring multiple transitions to other states.

However, runtime performance for a FSM system compares favourably to that of a BDI system, especially when the amount of Agents is increased significantly.

Our BDI model maintains a low complexity when compared to the FSM design, and adding more behaviours does not affect this. However, as noted, we have not included any information regarding the Agents commitment values or its priority values for plans; without this, Agents may react in a unrealistic manner. In the BDI system the computational cost rises as the amount of Agents increases.

In our combined model, the FSM method becomes significantly less complex, requiring only two states to handle our one “instinctive behaviour”. When the FSM is in its “Not Escaping” state it passes control to the BDI system, which is also less complex as it has fewer Desires. By keeping control of instinctive behaviours in the FSM, we are able to avoid the higher computation cost of BDI and react more quickly to situations where it is deemed necessary, without the requirement of deliberating over a BDI system's Desires and checking commitment and priority values.

5 Conclusion

While existing AI methodologies are capable of creating Agents that behave in a semi-realistic manner, these techniques are often very complex to implement or result in behaviours that can be clearly unrealistic. We have shown that using a novel hybrid and hierarchical combination of these methods we created a technique that brings the best features of existing methodologies while attempting to avoid their limitations. By analysing the different AI methodologies in our test scenario we have been able to present a case that demonstrates how our combination of techniques creates a simpler implementation that results in a more efficient and realistic simulation than using techniques individually.

References

1. Patel, P., Hexmoor, H.: Designing BOTs with BDI agents. In: International Symposium on Collaborative Technologies and Systems, Baltimore: s.n., pp. 180–186 (2009)
2. Li, Y., Musilek, P., Wyard-Scott, L.: Fuzzy logic in agent-based game design. In: Proceedings of the 2004 Annual Meeting of the North American Fuzzy Information Processing Society, pp. 734–739 (2004)
3. El Rhalibi, A., Merabti, M.: A Hybrid Fuzzy ANN System for Agent Adaptation in a First Person Shooter. International Journal of Computer Games Technology (2008)
4. Rao, A.S., Georgeff, P.M.: BDI Agents: From Theory to Practice. In: Proceedings of the First International Conference on Multi-Agent Systems, San Francisco: s.n. (1995)
5. Bratman, M.E.: Intentions, Plans and Practical Reason. Harvard University Press, Massachusetts (1987)
6. Bhargava, H.K., Branley Jr., W.C.: Simulating belief systems of autonomous agents. Decision Support Systems 14, 329–348 (1995)
7. Ingrand, F.F., Georgeff, M.P., Rao, A.S.: An architecture for Real-Time Reasoning and System Control. In: IEEE Expert: Intelligent Systems and Their Applications, pp. 34–44 (1992)

8. Braubach, L., Pokahr, A., Lamersdorf, W.: Jadex: A BDI Reasoning Engine. In: Multi-Agent Programming. s.l., vol. 15, pp. 149–174. Springer, US (2005)
9. Busetta, P., et al.: JACK - Components for intelligent agents in JAVA. In: AgentLink News Letter, January 2, pp. 2-5 (1999)
10. Yue, B., de-Byl, P.: The state of the art in game AI standardisation. In: Proceedings of the 2006 International Conference on Game Research and Development, pp. 41–46. Murdoch University, Perth (2006)
11. Bartish, A., Thevathayan, C.: BDI Agents for Game Development. In: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 2, pp. 668–669. ACM, Bologna (2002)
12. Fisher, M.: A survey of concurrent MetateM — The language and its applications. In: Gabbay, D.M., Ohlbach, H.J. (eds.) ICTL 1994. LNCS (LNAI), vol. 827, pp. 480–505. Springer, Heidelberg (1994)
13. Dennis, L.A., Farwer, B., Bordini, R.H., Fisher, M., Wooldridge, M.J.: A Common Semantic Basis for BDI Languages. In: Dastani, M.M., El Fallah Seghrouchni, A., Ricci, A., Winikoff, M. (eds.) ProMAS 2007. LNCS (LNAI), vol. 4908, pp. 124–139. Springer, Heidelberg (2008)

Coupling of Immune Algorithms and Game Theory in Multiobjective Optimization

Pawel Jarosz[1] and Tadeusz Burczynski[1,2]

[1] Cracow University of Technology, Institue of Computer Modelling,
Warszawska 24, 31-155 Cracow, Poland
pjarosz@pk.edu.pl

[2] Silesian University of Technology, Department for Strength of Materials and
Computational Mechanics, 18A Konarskiego str., 44-100 Gliwice
tb@polsl.pl

Abstract. Multiobjective optimization problems have been solved in recent years by several researchers using different kind of algorithms, among them genetic and evolutionary algorithms and artificial immune systems. The results obtained during these tests were satisfactory, but these researchers observed that there still is a need for new ideas for algorithms which will increase efficiency and at the same time decrease the computational effort. In this paper the idea of coupling of immune algorithms with game theory is presented. The authors take out the most important elements from the artificial immune system, such as clonal selection and suppression, and couple them with the idea of Nash equilibrium. The new approach and some preliminary tests and results are presented here.

1 Introduction

In the real world it is difficult to find a problem of which only one aspect is important. Usually we have to take into consideration two or even more objectives. In such cases traditional approaches do not lead to good solutions. There is a need for the creation of new algorithms and approaches. One way of searching for such approaches is the application of population-based methods. New kinds of evolutionary algorithms have been created over the pas 25 years. Schaffer was the first author who presented a new approach for multi-objective optimization the Vector Evaluated Genetic Algorithm ([1][2]). Since that time many new concepts have been implemented.

For a few years now, a new population-based approach for solving different problems called the Artificial Immune System has been presented. There were many proposals for using these algorithms for optimization. This approach has also been adopted for solving multiobjective optimization. The first such algorithm was presented in 2002 by Coello Coello and Cruz Cortez [3]. Later on several concepts based on immunology were presented in [4] [5] [6]. On the other hand for some years a new interesting approach based on connecting Nash theory and evolutionary algorithms in multiobjective optimization has developed ([7]).

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 500–507, 2010.
© Springer-Verlag Berlin Heidelberg 2010

2 Artificial Immune Systems

The natural immune system is mainly built from lymphocytes and its main goal is to protect the organism against pathogens (infectious foreign elements). Artificial Immune Systems appeared in the '90s as a new area in Computational Intelligence. Based on the rules that structure the immune system, scientists tried to generate new possibilities for solving different problems. The following three mechanisms were developed the most: immune networks, clonal selection and negative selection. The immune network is based on idea that all B-cells are connected together to cooperate in order to facilitate antigen recognition. The cells stimulate and suppress each other to stabilize the network.

In clonal selection optimization, the fitness function in immune terms is the affinity of antibody to antigen. Antibodies are solutions of a problem collected in the population. As in the real immune system, antibodies are cloned. The number of clones is proportional to its affinity to antigen (value of the fitness function). Clones are subjected to hypermutation. These ideas are extended by suppression. This type of algorithm was described for the first time by Wierzchon in [9].

Negative Selection is based on the assumption that the goal of the immune system learn is how to categorize cells as self or non-self. The system has the ability to react to unknown antigens at the same time not reacting to self cells. [10]

3 Game Theory

Game theory is a branch of mathematics applied to different sciences: economics, sociology, politics as well as engineering. The goal of game theory is to describe strategic situations, where parties try to make a decision based on others' behavior. Games can be divided into two groups: cooperative and non-cooperative games. The cooperative game is when the players should build compromises to get maximum payoff. The cooperation between players is allowed in such games. The biggest payoff for all playera can be gained only through cooperation between them. On the other hand, in non-cooperative games there is no possibility of cooperation and communication between the players is not allowed.

The most popular applications of game theory is to find equilibriums in games. In such an equilibrium the player has a strategy which is the best in the current situation. One of the most important theorems in game theory is Nash equilibrium, defined by J.F. Nash in 1951 [7]. Imagine there is G amount of players playing the same game. Each player optimizes his strategy following his payoff function and it is given that all other players' strategies are fixed (optimized for themselves). If there is no player who can improve his strategy, it means that players' strategies are Nash strategies and they correspond to Nash equilibrium.

Taking into account the properties of cooperative games and Nash equilibrium it can be stated that these concepts are similar to problems with many objectives.

4 Multiobjective Optimization

Generally, the idea of Multiobjective Optimization can be written as follows:

$$minimize f_i(x), i = 1, ..., m \text{ where } x = [x_1, x_2, ..., x_n] \text{ and } x_i \in (l(i), r(i)) \quad (1)$$

For such defined optimization problems, several solutions better or worse according to all criteria can be found. If for a particular solution there are no solutions which are superior in terms of all criteria, it can be said that such a solution is non-dominated. Others are dominated. The goal of the optimization process is to find a set of non-dominated solutions in a feasible area. This type of set is named the Pareto front.

The great challenge is to evaluate a set of solutions found by a particular method. This is strictly connected with the metrics which are used for algorithm evaluation. One of the features of the found set of solutions, is their distribution over the Pareto front. It is expected that they will be equally spaced and the front representation is smooth and uniform. The metric which defines such a goal is named *Spacing* and was proposed by Schott in [11]. Another criterion which should be taken into account, when evaluating the found optimal set of solutions, is the closeness to true Pareto front. Van Veldhuizen and Lamont in [12] have proposed a metric named *Generational Distance*. There are also other metrics for example, discreet metrics *Error Ratio* which counts the number of solutions which belong to the true Pareto front. We decided to use the two presented metrics as the most often used ones, and representing the features which should have the found Pareto front.

5 Description of the IMmune GAme Theory MultiObective (IMGAMO) Algorithm

The metaphors of game theory and immunology are used to solve problems of multiobjective optimization. Each player has its own objective (payoff function in the Nash equilibrium). Nash strategy for a particular player is the optimum solution for this player's problem remembering that other players also play their best strategies. The solution to the optimized problem consists of several parameters, each of them assigned to one of the players. Each player optimizes only his parameters (his strategy) taking the rest of them as constant. The rest of the parameters are set taking the best solutions from other players. Solutions from all players should establish the solution to the problem. All players use the immune algorithm to optimize their objectives.

5.1 Detailed Description of the Algorithm

Here are the most important assumptions of the IMGAMO algorithm: each player has his own fitness function (payoff function), each player has assigned part of the parameters of the solution (strategy of this player), the rest of parameters are set as constant taken from the best solutions from other players, all solutions are coded with real values, the result of the algorithm (the found Pareto front) is stored in the result_population, each player optimizes his use of the immune algorithm to optimize his objective.

Parameters of the algorithm: G - number of fitness functions in problem, *pop_size* - size of the population for each player, *clon_number* - maximum number of clones in clonal selection, *exchange_factor* - number of iteration when exchange parameters between player is performed, *iterations* - number of iterations of the algorithm.

The algorithm is executed as follows:

1. G players generate their populations which are initialized with size *pop_size*,
2. Empty *population_result* is created,
3. Each player has assigned one of the fitness functions,
4. Each parameter of the problem is randomly assigned to one of the player,
5. *iter*=0,
6. *exchange_iter*=0,
7. Each player run the *Clonal_selection* procedure:
   (a) calculate the fitness of the solution,
   (b) sort the population taking into account the fitness,
   (c) the best solution creates *clon_number* clones,
   (d) each next solution creates one clone less,
   (e) each clone is mutated (only parameters optimized by this player are mutated),
   (f) if the fitness of the mutated clone is better than the parent replace parent with clone,
8. Each player runs the *Suppression* procedure:
   (a) Each solution is checked for the number of solutions which are dominated (*dominated_solutions[i]*) by this one in the secondary population. If *dominated_solutions[i]>0* the solution is stored to the secondary population, and these dominated ones are removed,
   (b) calculate the average of dominated solutions *dominated_solutions_avg*,
   (c) If *domination_number[i]* <*dominated_solutions_avg* it is replaced by randomly generated one,
   (d) Additionally if the solution is nondominated in the secondary population and it was not added before, it is added,
9. If *exchange_iter=exchange_factor*,
   (a) Take the best solutions from each player,
   (b) store optimized parameters from each player to others populations as constant,
   (c) Each parameter of the problem is randomly assigned to one of the player
10. *Exchange_iter++*,
11. *iter++*,
12. if *iter* <*iterations* go to point 7,
13. The solution of the problem (Pareto front) is stored in population result.

Clonal selection is an element of the algorithm which is based on Nash equilibrium. Each player searches for optimal values of his parameters to get maximum payoff, taking the others' parameters as constants the best possible values from other players. During clonal selection the solutions which are best according to all criteria are found and they approach to the Pareto front. The goal of suppression is to diversify of the solutions in populations. The Pareto front has to fulfill two conditions. The first is to be as close as possible to the real front this function is performed by clonal selection. Secondly, it should be regularly distributed on this front. This is the task of the suppression.

6 Numerical Tests

In this article the results obtained by IMGAMO algorithm in two classical mutiobjective test problems are presented. In order to evaluate an algorithm, it is compared to the MISA algorithm one representative of state of the art immune algorithms for multiobjective optimization. For each problem the results are compared using two metrics: spacing and inverted generational distance. The parameters used in the experiments were adapted to evaluate fitness functions around 12000 times.

Table 1. The parameters for the IMISA and IMGAMO algorithms

MISA	*IMGAMO*
population_size: 100	iterations: 100
number of grid subdivisions = 25	clon_number: 15
size of the external population = 100	population_size: 100
	exchange_factor: 3

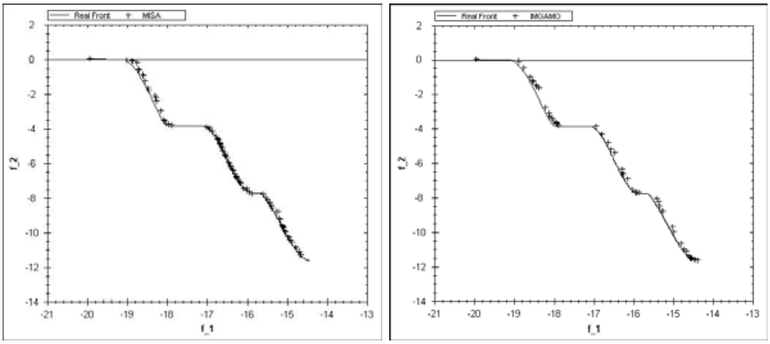


Fig. 1. Pareto fronts obtained using the MISA and IMGAMO algorithms for solving Kursave problem

6.1 The Kursave Problem

This is a two-objective three-parameter problem proposed by Kursave [13].

$$\begin{aligned} &\text{minimize} f_1(\overrightarrow{x}) = \sum_{i=1}^{n-1}\left(-10exp(-0.2\sqrt{x_i^2 + x_{i+1}^2})\right) \\ &\text{minimize} f_2(\overrightarrow{x}) = \sum_{i=1}^{n}(|x_i^{0.8}| + 5sin(x_i)^3) \end{aligned} \quad (2)$$

where

$5 \leq x_1, x_2, x_3 \leq 5$.

The true Pareto front is disconnected and consists of three curves. As can be seen in Figure 1 both algorithms found the front. MISA's front is closer to the true front (MISA G=0.0092, IMGAMO G=0,0148), but IMGAMO's front is better distributed (MISA S=0,1078, IMGAMO=0,0874).

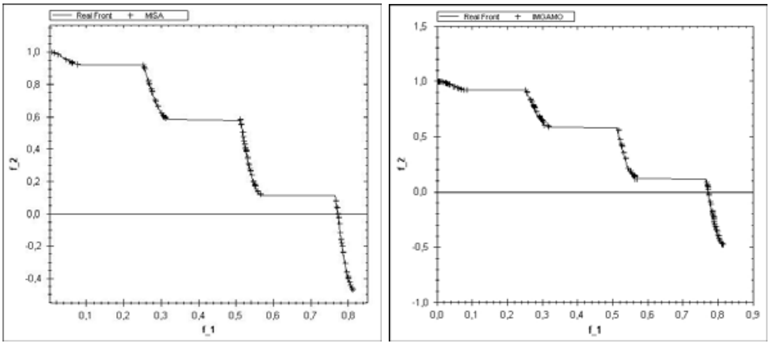


Fig. 2. Pareto fronts obtained using the MISA and IMGAMO algorithms for solving Deb problem

6.2 The Deb Problem

This problem was proposed by Deb [14] and the true Pareto front consists of four curves.

$$\begin{aligned} &\text{minimize} f_1(x, y) = x \\ &\text{minimize} f_2(x, y) = (1 + 10y)\left[1 - \left(\frac{x}{1+10y}\right)^{\alpha} - \frac{x}{1+10y}sin(2\pi qx)\right] \end{aligned} \quad (3)$$

where $0 \leq x, y \leq 1, q = 4, \alpha = 2$.

Table 2. The values of metrics for Kursave and Deb problems

MISA			*IMGAMO*		
	GD	S		GD	S
KursaveProblem					
Average	0,0092	0,1078	Average	0,0148	0,0874
Maximum	0,0340	0,1450	Maximum	0,0350	0,1500
Minimum	0,0044	0,0790	Minimum	0,0024	0,0150
Debproblem					
Average	0,0380	0,0847	Average	0,000079	0,001135
Maximum	0,1350	0,2090	Maximum	0,000240	0,003200
Minimum	0,0004	0,0075	Minimum	0,000013	0,000480

In this problem, both algorithms found all segments of the front and it is very close to the true front and well distributed (Figure 2). Looking at the metrics IMGAMO perform better in this problem. Both metrics are significantly lower for the IMGAMO algorithm.

7 Conclusions and Future Work

A new approach to multiobjective optimization was presented in this paper. The novel idea of coupling ideas from Artificial Immune Systems and Game theory was adopted to solve problems with many objectives. The IMGAMO (IMmune GAme theory MultiObjective) algorithm was described and the results of tests on traditional problems were presented. The results of the algorithm behavior were compared with that of the state-of-the-art immune algorithm MISA. The first tests and results showed that there are cases were IMGAMO and MISA behave similarly, but there are also problems, where IMGAMO performs significantly better. The results presented in this paper are quite satisfactory and promising. It is important to test this algorithm in large-scale problems with a larger number of objectives and variables. The new ideas for the suppression should be tested. Additionally the algorithm should be adopted to solve constrained problems. This is a very important issue in multiobjective optimization, and up to now IMGAMO has not dealt with such problems.

Acknowledgments

This work was supported by the Polish Ministry of Science and Higher Education under grant No. N N519 405437.

References

1. Schaffer, J.D.: Some experiments in machine learning using vector evaluated genetic algorithms (artificial intelligence, optimization, adaptation, pattern recognition). PhD thesis, Nashville, TN, USA (1984)
2. Schaffer, J.D.: Multiple objective optimization with vector evaluated genetic algorithms. In: Proceedings of the 1st International Conference on Genetic Algorithms, Hillsdale, NJ, USA, pp. 93–100. L. Erlbaum Associates Inc., Mahwah (1985)
3. Coello, C.A., Cortés, N.C.: Solving multiobjective optimization problems using an artificial immune system. Genetic Programming and Evolvable Machines 6(2), 163–190 (2005)
4. Gong, M., Jiao, L., Du, H., Bo, L.: Multiobjective immune algorithm with nondominated neighbor-based selection. Evol. Comput. 16(2), 225–255 (2008)
5. Gao, J., Wang, J.: Wbmoais: A novel artificial immune system for multiobjective optimization. Comput. Oper. Res. 37(1), 50–61 (2010)
6. Luh, G.C., Chueh, C.H., Liu, W.W.: MOIA: Multi-Objective Immune Algorithm. Engineering Optimization 35(2), 143–164 (2003)
7. Sefrioui, M., Periaux, J.: Nash genetic algorithms: Examples and applications. In: Proceedings of the 2000 Congress on Evolutionary Computation CEC 2000, La Jolla Marriott Hotel La Jolla, California, USA, June-September 2000, pp. 509–516. IEEE Press, Los Alamitos (2000)
8. Jarosz, P., Burczynski, T.: Immune algorithm for multi-modal optimization - numerical tests in intelligent searching. Recent Developments in Artificial Intelligence Methods (2004)
9. Wierzchon, S.T.: Function optimization by the immune metaphor. Task Quarterly 6 (2002)
10. Dasgupta, D.: Advances in artificial immune systems. IEEE Computational Intelligence Magazine 1(4), 40–49 (2006)
11. Schott, J.R.: Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts (May 1995)
12. Van Veldhuizen, D., Lamont, G.: On measuring multiobjective evolutionary algorithm performance. In: Proceedings of the 2000 Congress on Evolutionary Computation, vol. 1, pp. 204–211 (2000)
13. Kursawe, F.: A variant of evolution strategies for vector optimization. In: PPSN I: Proceedings of the 1st Workshop on Parallel Problem Solving from Nature, London, UK, pp. 193–197. Springer, Heidelberg (1991)
14. Deb, K.: Multi-objective genetic algorithms: Problem difficulties and construction of test problems. Evol. Comput. 7(3), 205–230 (1999)
15. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary Computation 8, 173–195 (2000)

Intelligent E-Learning Systems for Evaluation of User's Knowledge and Skills with Efficient Information Processing

Wojciech Kacalak[1], Maciej Majewski[1], and Jacek M. Zurada[2]

[1] Koszalin University of Technology, Faculty of Mechanical Engineering
Raclawicka 15-17, 75-620 Koszalin, Poland
{wojciech.kacalak,maciej.majewski}@tu.koszalin.pl

[2] University of Louisville, Department of Electrical and Computer Engineering
405 Lutz Hall, Louisville, KY 40292, USA
jacek.zurada@louisville.edu

Abstract. This paper presents a new concept of an e-learning system with intelligent two-way speech communication between the system and its users. Computational intelligence methods allow for analysis, evaluation and assessment of user's knowledge and skills and user's ability for efficient information processing. The system is also capable of control, supervision and optimization of the e-learning process. Developed as a prototype for mobile technologies, the communication system by speech and natural language between the e-learning system and external users consists of biometric user identification, speech recognition, word and sentence recognition, sentence meaning analysis, and user reaction assessment. Also discussed are selected new concepts of intelligent e-learning systems using speech communication. The discussion focuses on recognition and evaluation of spoken natural language sentences with hybrid neural networks.

Keywords: e-learning system, user-computer interaction, speech interface, artificial intelligence, mobile technology, cybernetics.

1 Introduction

Intelligent mobile systems for implementation, control, supervision and optimization of distance learning when users communicate with an e-learning system by speech and natural language are becoming increasingly important. These systems enable remote supervision of the e-learning process quality using spoken commands in natural language and allow for quality learning. The described remote system is an innovative solution making it possible to better exploit current e-learning methods. The presented solution can be used to create standard of intelligent mobile systems for implementation, control, supervision and optimization of e-learning processes based on two-way communication by speech and natural language between the e-learning system and the user.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 508–515, 2010.
© Springer-Verlag Berlin Heidelberg 2010

There are several advantages of intelligent e-learning systems based on intelligent two-way communication by speech and natural language between the e-learning system and the user. They are robust against user's errors and an efficient implementation, control, supervision and optimization of the e-learning process. They also improve the co-operation between a user and an e-learning system in respect to the richness of communication. E-learning decision and optimization systems can be remote elements with regard to an e-learning system.

2 The State of the Art

Design and implementation of spoken language understanding systems is an important field of research [2,7,9]. Vocabularies for these systems are usually about tens of thousands of words, and they recognize spontaneous speech and natural language. Current research in spoken language systems focuses on understanding of spoken input, and most applications involve a collaboration between the human and the computer. In many cases, spoken language output is an appropriate means of communication that may or may not be taken advantage of, because of lack of coordination of understanding components with system outputs [5]. This paper proposes an approach to deal with the above problem.

E-learning systems offer new possibilities in learning, because a user can get immediate feedback on solutions to problems, learning paths can be individualized, etc. At present, numerous e-learning tools with varying functionality and purposes exist [1,3,4]. E-learning is an alternative concept to the traditional tutoring system. E-learning forms a new way to empower a workforce with the necessary skills and knowledge [8]. Towards this goal, various e-learning systems have been developed during the last years, however, most of them are old-fashioned applications without advanced capabilities which could be delivered with artificial intelligence methods [10,11].

3 A New Concept of Intelligent E-Learning Systems

The new concept involves e-learning systems supported by the artificial intelligence methods and equipped with two-way speech communication between the e-learning system and the user (Fig. 1A). They perform analysis, evaluation and assessment of the user's knowledge and skills and can efficiently process communication between the user and e-learning systems. The system is also capable of control, supervision and optimization of the e-learning process. It is equipped with hybrid artificial neural networks and intelligent two-way voice communication which can optionally use mobile technology (Fig. 1B). The communication via spoken natural language relies on biometric user identification, and performs speech recognition, word and sentence recognition, sentence meaning analysis, and user reaction assessment. The intelligent e-learning system (Fig. 2A) has been developed for personal computers as well as mobile technology devices (Fig. 2B).

The complete intelligent e-learning system in expanded form is shown in Fig. 3. The numbers in the cycle represent the successive phases of information processing. The authenticated user's spoken utterance in natural language is processed for recognition of words and sentences through a cycle of sentence meaning analysis (Fig. 4A) [5,6]. The Hamming neural networks classify words and sentences

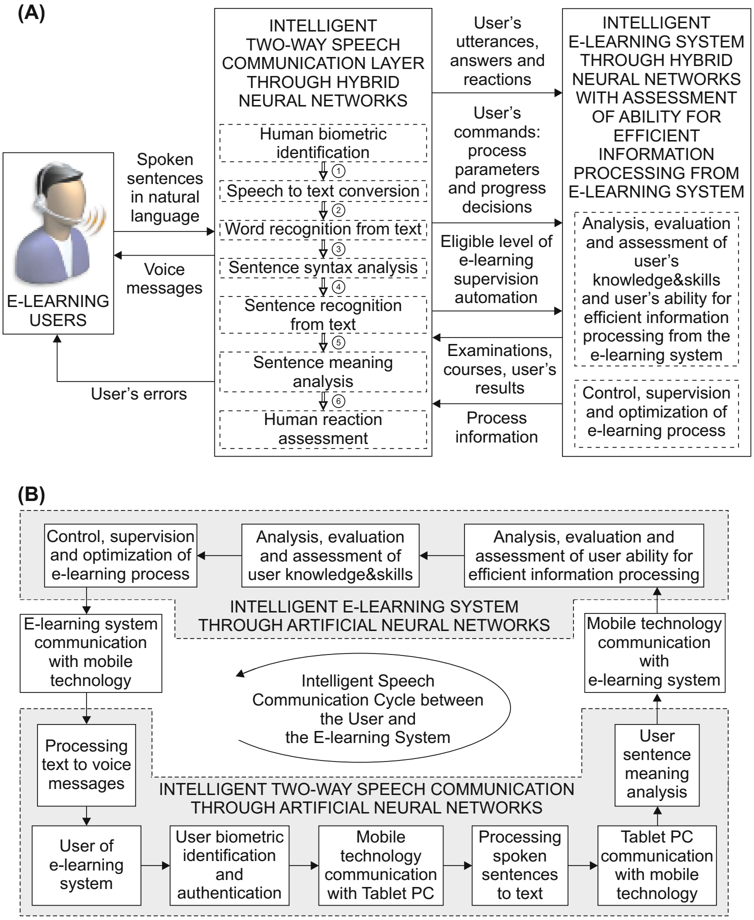
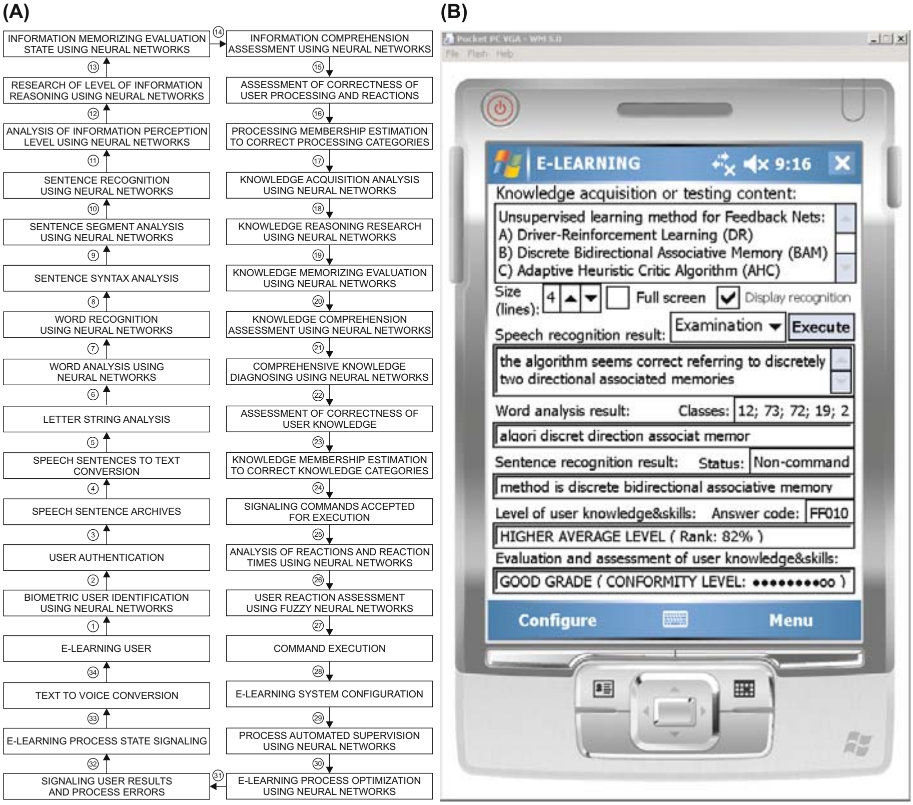

(A)
E-LEARNING USERS
Spoken sentences in natural language
Voice messages
User's errors
INTELLIGENT TWO-WAY SPEECH COMMUNICATION LAYER THROUGH HYBRID NEURAL NETWORKS
Human biometric identification
1
Speech to text conversion
2
Word recognition from text
3
Sentence syntax analysis
4
Sentence recognition from text
5
Sentence meaning analysis
6
Human reaction assessment
User's utterances, answers and reactions
User's commands: process parameters and progress decisions
Eligible level of e-learning supervision automation
Examinations, courses, user's results
Process information
INTELLIGENT E-LEARNING SYSTEM THROUGH HYBRID NEURAL NETWORKS WITH ASSESSMENT OF ABILITY FOR EFFICIENT INFORMATION PROCESSING FROM E-LEARNING SYSTEM
Analysis, evaluation and assessment of user's knowledge&skills and user's ability for efficient information processing from the e-learning system
Control, supervision and optimization of e-learning process
(B)
Control, supervision and optimization of e-learning process
Analysis, evaluation and assessment of user knowledge&skills
Analysis, evaluation and assessment of user ability for efficient information processing
INTELLIGENT E-LEARNING SYSTEM THROUGH ARTIFICIAL NEURAL NETWORKS
E-learning system communication with mobile technology
Mobile technology communication with e-learning system
Intelligent Speech Communication Cycle between the User and the E-learning System
Processing text to voice messages
User sentence meaning analysis
INTELLIGENT TWO-WAY SPEECH COMMUNICATION THROUGH ARTIFICIAL NEURAL NETWORKS
User of e-learning system
User biometric identification and authentication
Mobile technology communication with Tablet PC
Processing spoken sentences to text
Tablet PC communication with mobile technology

Fig. 1. (A) Concept of an intelligent e-learning system with speech communication, (B) e-learning system for mobile technology

and identify their meaning (fig. 4B) [6]. The recognized meaningful sentences are further processed for analysis, evaluation and assessment of the user's ability for efficient processing of information streams from e-learning systems. The user's ability is appraised in respect of perception, analyzing and reasoning of information, and also information memorizing and comprehension. The result is the assessment of the user's information processing and reactions. Next, the user's utterance or answer is analyzed and evaluated in terms of knowledge and skills. The analyses and evaluations include knowledge acquisition and reasoning as well as knowledge memorizing and comprehension. The system diagnoses comprehensive knowledge and consecutively assesses correctness of the user's knowledge and skills. Further, it assesses the user's knowledge and skills and executes the user's commands. The e-learning system is capable of supervision and optimization of the process and two-way speech communication with the user.


(A)
E-LEARNING USER
1
BIOMETRIC USER IDENTIFICATION USING NEURAL NETWORKS
2
USER AUTHENTICATION
3
SPEECH SENTENCE ARCHIVES
4
SPEECH SENTENCES TO TEXT CONVERSION
5
LETTER STRING ANALYSIS
6
WORD ANALYSIS USING NEURAL NETWORKS
7
WORD RECOGNITION USING NEURAL NETWORKS
8
SENTENCE SYNTAX ANALYSIS
9
SENTENCE SEGMENT ANALYSIS USING NEURAL NETWORKS
10
SENTENCE RECOGNITION USING NEURAL NETWORKS
11
ANALYSIS OF INFORMATION PERCEPTION LEVEL USING NEURAL NETWORKS
12
RESEARCH OF LEVEL OF INFORMATION REASONING USING NEURAL NETWORKS
13
INFORMATION MEMORIZING EVALUATION STATE USING NEURAL NETWORKS
14
INFORMATION COMPREHENSION ASSESSMENT USING NEURAL NETWORKS
15
ASSESSMENT OF CORRECTNESS OF USER PROCESSING AND REACTIONS
16
PROCESSING MEMBERSHIP ESTIMATION TO CORRECT PROCESSING CATEGORIES
17
KNOWLEDGE ACQUISITION ANALYSIS USING NEURAL NETWORKS
18
KNOWLEDGE REASONING RESEARCH USING NEURAL NETWORKS
19
KNOWLEDGE MEMORIZING EVALUATION USING NEURAL NETWORKS
20
KNOWLEDGE COMPREHENSION ASSESSMENT USING NEURAL NETWORKS
21
COMPREHENSIVE KNOWLEDGE DIAGNOSING USING NEURAL NETWORKS
22
ASSESSMENT OF CORRECTNESS OF USER KNOWLEDGE
23
KNOWLEDGE MEMBERSHIP ESTIMATION TO CORRECT KNOWLEDGE CATEGORIES
24
SIGNALING COMMANDS ACCEPTED FOR EXECUTION
25
ANALYSIS OF REACTIONS AND REACTION TIMES USING NEURAL NETWORKS
26
USER REACTION ASSESSMENT USING FUZZY NEURAL NETWORKS
27
COMMAND EXECUTION
28
E-LEARNING SYSTEM CONFIGURATION
29
PROCESS AUTOMATED SUPERVISION USING NEURAL NETWORKS
30
E-LEARNING PROCESS OPTIMIZATION USING NEURAL NETWORKS
31
SIGNALING USER RESULTS AND PROCESS ERRORS
32
E-LEARNING PROCESS STATE SIGNALING
33
TEXT TO VOICE CONVERSION
34
(B)
Pocket PC VGA - WM 5.0
File Flash Help
E-LEARNING
9:16
Configure
Menu

Knowledge acquisition or testing content:

Unsupervised learning method for Feedback Nets:
A) Driver-Reinforcement Learning (DR)
B) Discrete Bidirectional Associative Memory (BAM)
C) Adaptive Heuristic Critic Algorithm (AHC)

Size (lines): 4 □ Full screen ☑ Display recognition

Speech recognition result: Examination Execute

the algorithm seems correct referring to discretely two directional associated memories

Word analysis result: Classes: 12; 73; 72; 19; 2

algori discret direction associat memor

Sentence recognition result: Status: Non-command

method is discrete bidirectional associative memory

Level of user knowledge&skills: Answer code: FF010

HIGHER AVERAGE LEVEL (Rank: 82%)

Evaluation and assessment of user knowledge&skills:

GOOD GRADE (CONFORMITY LEVEL: ●●●●●●●●○○)

Fig. 2. (A) Complete intelligent e-learning system, (B) prototype of the intelligent e-learning system for mobile technology: an example question and answer by speech and natural language and the user knowledge evaluation and assessment

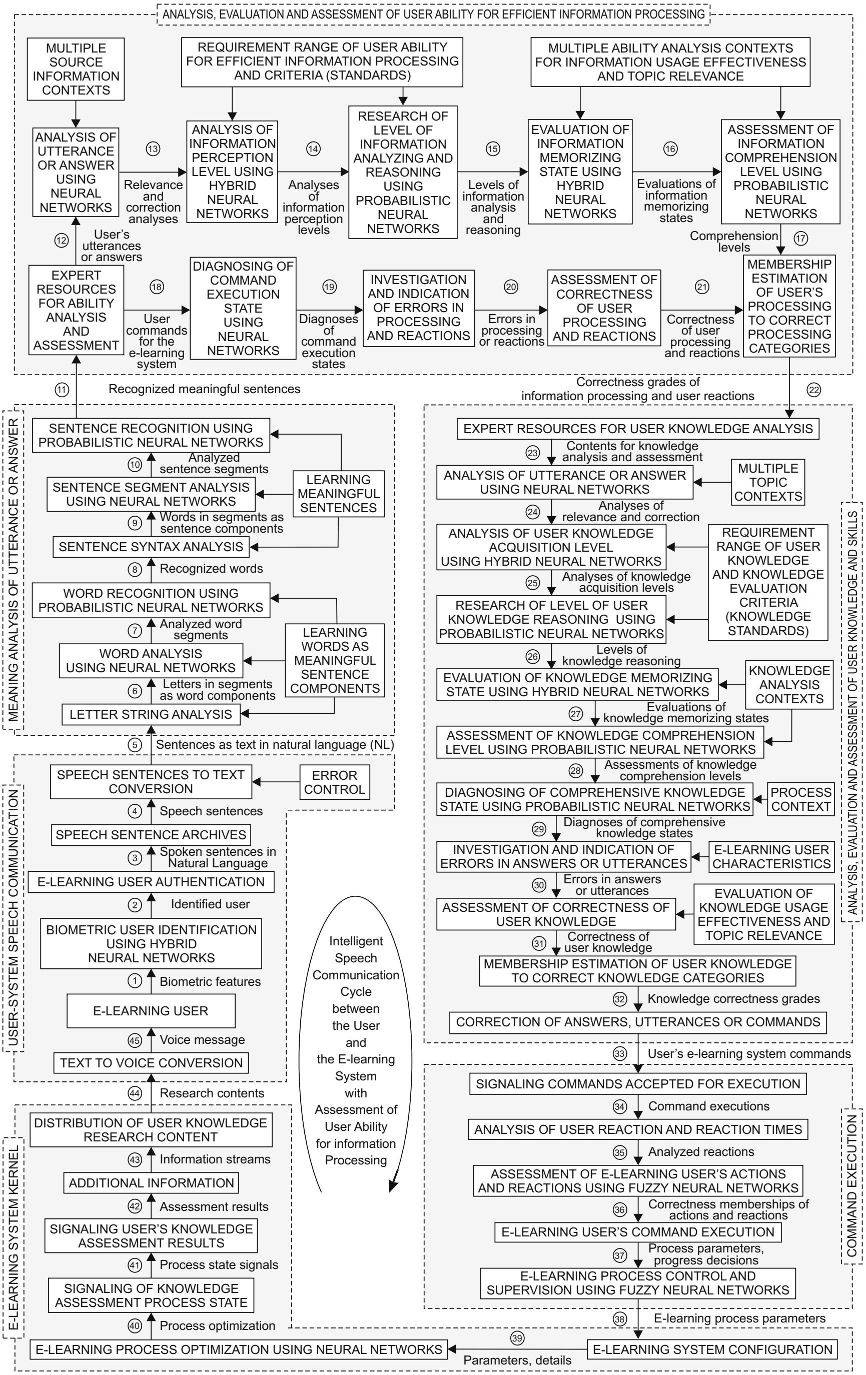

ANALYSIS, EVALUATION AND ASSESSMENT OF USER ABILITY FOR EFFICIENT INFORMATION PROCESSING
MULTIPLE SOURCE INFORMATION CONTEXTS
REQUIREMENT RANGE OF USER ABILITY FOR EFFICIENT INFORMATION PROCESSING AND CRITERIA (STANDARDS)
MULTIPLE ABILITY ANALYSIS CONTEXTS FOR INFORMATION USAGE EFFECTIVENESS AND TOPIC RELEVANCE
ANALYSIS OF UTTERANCE OR ANSWER USING NEURAL NETWORKS
13
Relevance and correction analyses
ANALYSIS OF INFORMATION PERCEPTION LEVEL USING HYBRID NEURAL NETWORKS
14
Analyses of information perception levels
RESEARCH OF LEVEL OF INFORMATION ANALYZING AND REASONING USING PROBABILISTIC NEURAL NETWORKS
15
Levels of information analysis and reasoning
EVALUATION OF INFORMATION MEMORIZING STATE USING HYBRID NEURAL NETWORKS
16
Evaluations of information memorizing states
ASSESSMENT OF INFORMATION COMPREHENSION LEVEL USING PROBABILISTIC NEURAL NETWORKS
Comprehension levels
17
12
User's utterances or answers
EXPERT RESOURCES FOR ABILITY ANALYSIS AND ASSESSMENT
18
User commands for the e-learning system
DIAGNOSING OF COMMAND EXECUTION STATE USING NEURAL NETWORKS
19
Diagnoses of command execution states
INVESTIGATION AND INDICATION OF ERRORS IN PROCESSING AND REACTIONS
20
Errors in processing or reactions
ASSESSMENT OF CORRECTNESS OF USER PROCESSING AND REACTIONS
21
Correctness of user processing and reactions
MEMBERSHIP ESTIMATION OF USER'S PROCESSING TO CORRECT PROCESSING CATEGORIES
11
Recognized meaningful sentences
Correctness grades of information processing and user reactions
22
MEANING ANALYSIS OF UTTERANCE OR ANSWER
SENTENCE RECOGNITION USING PROBABILISTIC NEURAL NETWORKS
10
Analyzed sentence segments
SENTENCE SEGMENT ANALYSIS USING NEURAL NETWORKS
LEARNING MEANINGFUL SENTENCES
9
Words in segments as sentence components
SENTENCE SYNTAX ANALYSIS
8
Recognized words
WORD RECOGNITION USING PROBABILISTIC NEURAL NETWORKS
7
Analyzed word segments
WORD ANALYSIS USING NEURAL NETWORKS
LEARNING WORDS AS MEANINGFUL SENTENCE COMPONENTS
6
Letters in segments as word components
LETTER STRING ANALYSIS
5
Sentences as text in natural language (NL)
USER-SYSTEM SPEECH COMMUNICATION
SPEECH SENTENCES TO TEXT CONVERSION
ERROR CONTROL
4
Speech sentences
SPEECH SENTENCE ARCHIVES
3
Spoken sentences in Natural Language
E-LEARNING USER AUTHENTICATION
2
Identified user
BIOMETRIC USER IDENTIFICATION USING HYBRID NEURAL NETWORKS
1
Biometric features
E-LEARNING USER
45
Voice message
TEXT TO VOICE CONVERSION
44
Research contents
E-LEARNING SYSTEM KERNEL
DISTRIBUTION OF USER KNOWLEDGE RESEARCH CONTENT
43
Information streams
ADDITIONAL INFORMATION
42
Assessment results
SIGNALING USER'S KNOWLEDGE ASSESSMENT RESULTS
41
Process state signals
SIGNALING OF KNOWLEDGE ASSESSMENT PROCESS STATE
40
Process optimization
E-LEARNING PROCESS OPTIMIZATION USING NEURAL NETWORKS
Intelligent Speech Communication Cycle between the User and the E-learning System with Assessment of User Ability for information Processing
EXPERT RESOURCES FOR USER KNOWLEDGE ANALYSIS
23
Contents for knowledge analysis and assessment
ANALYSIS OF UTTERANCE OR ANSWER USING NEURAL NETWORKS
MULTIPLE TOPIC CONTEXTS
24
Analyses of relevance and correction
ANALYSIS OF USER KNOWLEDGE ACQUISITION LEVEL USING HYBRID NEURAL NETWORKS
REQUIREMENT RANGE OF USER KNOWLEDGE AND KNOWLEDGE EVALUATION CRITERIA (KNOWLEDGE STANDARDS)
25
Analyses of knowledge acquisition levels
RESEARCH OF LEVEL OF USER KNOWLEDGE REASONING USING PROBABILISTIC NEURAL NETWORKS
26
Levels of knowledge reasoning
EVALUATION OF KNOWLEDGE MEMORIZING STATE USING HYBRID NEURAL NETWORKS
KNOWLEDGE ANALYSIS CONTEXTS
27
Evaluations of knowledge memorizing states
ASSESSMENT OF KNOWLEDGE COMPREHENSION LEVEL USING PROBABILISTIC NEURAL NETWORKS
28
Assessments of knowledge comprehension levels
DIAGNOSING OF COMPREHENSIVE KNOWLEDGE STATE USING PROBABILISTIC NEURAL NETWORKS
PROCESS CONTEXT
29
Diagnoses of comprehensive knowledge states
INVESTIGATION AND INDICATION OF ERRORS IN ANSWERS OR UTTERANCES
E-LEARNING USER CHARACTERISTICS
30
Errors in answers or utterances
ASSESSMENT OF CORRECTNESS OF USER KNOWLEDGE
EVALUATION OF KNOWLEDGE USAGE EFFECTIVENESS AND TOPIC RELEVANCE
31
Correctness of user knowledge
MEMBERSHIP ESTIMATION OF USER KNOWLEDGE TO CORRECT KNOWLEDGE CATEGORIES
32
Knowledge correctness grades
CORRECTION OF ANSWERS, UTTERANCES OR COMMANDS
ANALYSIS, EVALUATION AND ASSESSMENT OF USER KNOWLEDGE AND SKILLS
33
User's e-learning system commands
SIGNALING COMMANDS ACCEPTED FOR EXECUTION
34
Command executions
ANALYSIS OF USER REACTION AND REACTION TIMES
35
Analyzed reactions
ASSESSMENT OF E-LEARNING USER'S ACTIONS AND REACTIONS USING FUZZY NEURAL NETWORKS
36
Correctness memberships of actions and reactions
E-LEARNING USER'S COMMAND EXECUTION
37
Process parameters, progress decisions
E-LEARNING PROCESS CONTROL AND SUPERVISION USING FUZZY NEURAL NETWORKS
COMMAND EXECUTION
38
E-learning process parameters
39
E-LEARNING SYSTEM CONFIGURATION
Parameters, details

Fig. 3. Detailed complete intelligent e-learning system

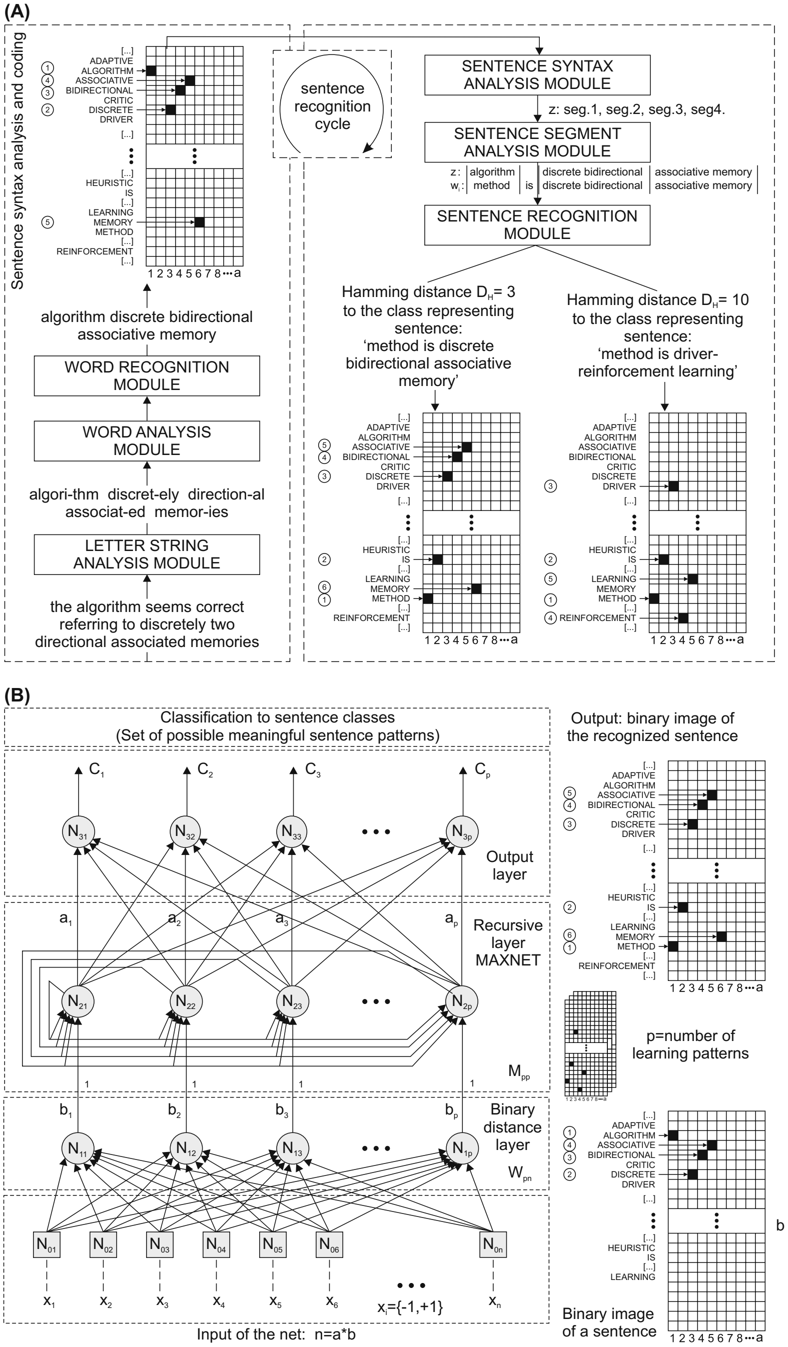

(A)
Sentence syntax analysis and coding
[...]
ADAPTIVE
ALGORITHM
ASSOCIATIVE
BIDIRECTIONAL
CRITIC
DISCRETE
DRIVER
[...]
HEURISTIC
IS
LEARNING
MEMORY
METHOD
REINFORCEMENT
1 2 3 4 5 6 7 8 ··· a
algorithm discrete bidirectional associative memory
WORD RECOGNITION MODULE
WORD ANALYSIS MODULE
algori-thm discret-ely direction-al associat-ed memor-ies
LETTER STRING ANALYSIS MODULE
the algorithm seems correct referring to discretely two directional associated memories
sentence recognition cycle
SENTENCE SYNTAX ANALYSIS MODULE
z: seg.1, seg.2, seg.3, seg4.
SENTENCE SEGMENT ANALYSIS MODULE
z: | algorithm | | discrete bidirectional | associative memory
w_i: | method | is | discrete bidirectional | associative memory
SENTENCE RECOGNITION MODULE
Hamming distance D_H= 3 to the class representing sentence: 'method is discrete bidirectional associative memory'
Hamming distance D_H= 10 to the class representing sentence: 'method is driver-reinforcement learning'
1
2
3
4
5
6
(B)
Classification to sentence classes (Set of possible meaningful sentence patterns)
C_1
C_2
C_3
C_p
N_31
N_32
N_33
N_3p
Output layer
a_1
a_2
a_3
a_p
Recursive layer MAXNET
N_21
N_22
N_23
N_2p
M_pp
1
b_1
b_2
b_3
b_p
Binary distance layer
N_11
N_12
N_13
N_1p
W_pn
N_01
N_02
N_03
N_04
N_05
N_06
N_0n
x_1
x_2
x_3
x_4
x_5
x_6
x_i={-1,+1}
x_n
Input of the net: n=a*b
Output: binary image of the recognized sentence
p=number of learning patterns
b
Binary image of a sentence

Fig. 4. (A) Meaning analysis cycle for user's utterance or answer, (B) neural network for sentence recognition

4 Experimental Results

The prototype of the intelligent communication system was developed for Windows Mobile for a PDA (Pocket PC) and a Tablet PC for Windows.

As shown in Fig. 5A, the ability of the implemented neural network to recognize a word depends on the number of letters of that word. For best performance, the neural network requires a minimum number of letters of each word being recognized as its input.

The ability of the neural network to recognize the sentence depends on the sentence length as shown in Fig. 5B. Similarly, for best sentence recognition, the neural network requires a certain minimum wordcount of the given sentence.

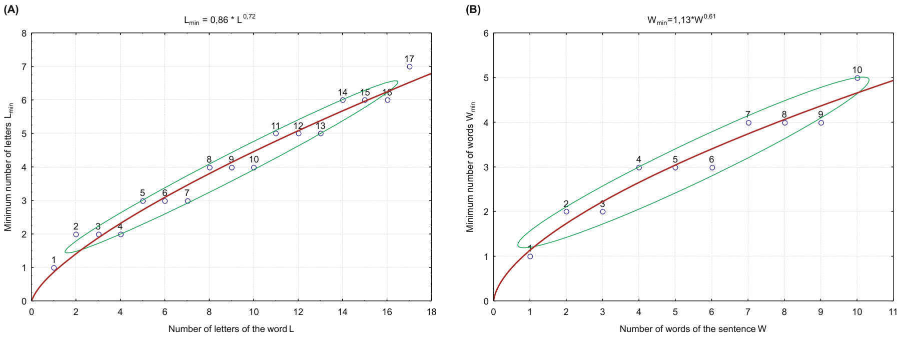


Fig. 5. Experimental results [5]: (A) sensitivity of spoken word recognition: minimum number of letters of the word being recognized vs. number of word letters, (B) sensitivity of spoken sentence meaning recognition: minimum number of words of the sentence being recognized vs. number of sentence component words

The experimental implementation of the intelligent e-learning system prototype allowed for its evaluation. The testing has allowed to determine the following attributes of the system [1,8]:

- Suitability, expressing the degree of appropriateness of the system to the tasks which have to be accomplished;
- Learnability, which conveys how easy it is for the user to learn the system and how rapidly the user can begin to work with it;
- Error rate, which reflects the error ratio while working with the system.

5 Conclusions and Perspectives

A speech interface between users and e-learning systems using the natural language is ideal because it is the most natural, flexible, efficient, and economical form of human communication. Application of hybrid neural networks has allowed recognition of sentences of similar meanings but of different lexical and grammatical patterns, which will undoubtedly be the most important for communication between humans and computer systems. This approach also enriches the methods of natural language processing.

The effectiveness of the e-learning system increases if it implements the meaning analysis of the user's spoken utterance, as well as analysis, evaluation and assessment of the user's knowledge, skills and ability for efficient information processing. In the e-learning systems, the condition of high quality communication between users and e-learning systems is the analysis of the e-learning process state before an utterance is performed by the user, and use of artificial intelligence for the analysis, evaluation and assessment of the answer or command.

The developed flexible intelligent e-learning system can be extended to various applications. The experimental results of the proposed system show its promising performance for further development and experiments. The system described in this paper is a conceptually new approach to the problem of coordination of understanding components with system outputs in a spoken language system.

References

1. Ardito, C., Costabile, M.F., De Marsico, M., Lanzilotti, R., Levialdi, S., Roselli, T., Rossano, V.: An approach to usability evaluation of e-learning applications. Universal Access in the Information Society 4(3), 270–283 (2006)
2. Ayres, T., Nolan, B.: Voice activated command and control with speech recognition over WiFi. Science of Computer Programming 59(1-2), 109–126 (2006)
3. Govindasamy, T.: Successful implementation of e-Learning. The Internet and Higher Education 4(3-4), 287–299 (2002)
4. Ismail, J.: The design of an e-learning system. The Internet and Higher Education 4(3-4), 329–336 (2002)
5. Kacalak, W., Majewski, M.: E-learning systems with artificial intelligence in engineering. In: Huang, D.-S., Jo, K.-H., Lee, H.-H., Kang, H.-J., Bevilacqua, V. (eds.) ICIC 2009. LNCS, vol. 5754, pp. 918–927. Springer, Heidelberg (2009)
6. Majewski, M., Zurada, J.M.: Sentence recognition using artificial neural networks. Knowledge-Based Systems 21(7), 629–635 (2008)
7. Paek, T., Chickering, D.M.: Improving command and control speech recognition on mobile devices: using predictive user models for language modeling. User Modeling and User-Adapted Interaction 17(1-2), 93–117 (2007)
8. Rosenberg, M.J.: E-Learning: strategies for delivering knowledge in the digital age. McGraw-Hill, New York (2001)
9. Wald, M., Bain, K.: Universal access to communication and learning: the role of automatic speech recognition. Universal Access in the Information Society 6(4), 435–447 (2008)
10. Zurada, J.M.: Introduction to Artificial Neural Systems. PWS Publishing Company, Boston (1992)
11. Zurada, J.M., Marks, R.J., Robinson, C.J.: Computational Intelligence: Imitating Life. IEEE Press, New York (1994)

Interactive Cognitive-Behavioral Decision Making System

Zdzisław Kowalczuk and Michał Czubenko

Gdansk University of Technology, Narutowicza 11/12, Gdansk, Poland

Abstract. The paper reports results of transforming a human psychological model into an interactive cognitive-behavioral, emotionally-driven system of making decisions conditioned both by the environment and the actual extend of the fulfillment of needs. Human psychological model is based on cognitive and personality psychology. It contains emotions, needs and a structure of human cognition processes. The effect of transformation is expressed with use of fuzzy-sets and neural-fuzzy networks.

1 Introduction

Creating a system functioning in a human-like way, has long been a principal subject of robotics. Taking into account the external mechanical aspect, a great number of artificial creatures (or robots) have been constructed, like antropoidal tramping robots. Whereas, considering an inner aspect, the well-known artificial neural networks have been conceived and applied. The concept of internet chatter-{ro}bots is widely exercised. Recently, a cat brain simulation have already been accomplished [1]. All these petite steps are made towards creating an artificial humanoid, synthetic organism, or android robot being designed to look and act like a human (at least from some point of view, like speaking actroids).

In practice, though, modeling and simulation of the human psychological aspects have been barely approached. There are some attempts to show emotions (or rather emote like actroids or avatars), they are not, however, those proper emotions that have been defined by psychology as the psychological states related to feelings (sentiments), thoughts, and behavior [2]. Such "psychological" systems can only give their external visualizations certain programmed features (like "smile" or "sadness"). Motivation has attracted even less attention. There are no systems having own, autonomous needs and willingness.

A system able to imitate human motivation could be functional in many circumstances, from security guards, smart devices, human artificial limbs, to personal companions. A security guard can be implemented in a form of a smart dog snooping around and looking for a thread to the integrity of a given sphere. A group of such guard-bots can co-operate and obey the orders of their supervisor. For esthetic reasons, available hand bio-manipulators look quite artificial. Thus apart from being managed by the disabled, they can also be mastered by adding certain natural-like spontaneous movements controlled by a motivation system in time of idleness. The perspectives of using different personal companions or servants appears to be limitless.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 516–523, 2010.

© Springer-Verlag Berlin Heidelberg 2010

Inventing an emotionally-driven system could be considered as a milestone on our way to designing and creating semi-autonomous robots. It was dreamed for centuries. Myths about machines-servants forged by Hefajstos are known from Antiquity. Middle Ages have their intelligent golems made of clay. Later on the novels of Asimow strongly affected our vision of autonomous robots.

The main idea is to model human psychology by prying about the nature in order to create a system with an ability for auto-adaptation. Such systems can be used for building different intelligent robots, from mobile autonomous units to multiagent systems.

2 Outline of Cognitive Science and Personality Psychology

A basis for creating autonomous systems based on human psychology can be found in both cognitive science and personality psychology. The former is responsible for active information processing, whereas the latter considers generating motivations for the system reaction.

2.1 Cognitive Processes and Their Basic Model

In general, the cognitive approach to the decision-making processes postulates that the knowledge being a basis for decisions is not simply created by passive accumulation and storage of data. Instead, an active processing of the data takes place. This means that the structures of the human cognitive processes constitute a solid basis for modeling the decision process of thinking entities.

A most complex model, having a deep portrayal in reality, is shown in Fig. 1, which depicts the human cognitive processes and the ways they inter-communicate. The processes can be elementary or composed. Perception, attention, and memory[1] are the elementary processes. Thinking and language[2] are the composed processes [3]. We accept here the classical, although some authors consider the language within the thinking processes, whereas others also isolate calculating, abstracting, differentiating, creating ideas, formulating judgments, making decisions, organizing and planning, as well as solving problems [4].

The path of information starts when a stimulus is received, and terminates when a decision concerning a recognized problem is made. After receiving and pre-filtering a stimulus (the sensory perception), the discovery (perception) is responsible for the primal information processing consisting in the discovery of objects features and encoding them into the form of impressions. The impressions are then translated to perceptions, being a group of impressions concerning one object. The perceptions then are filtered, and known features are detected (the

[1] It is worth noticing that the memory can be treated as a container of data, or as a process of storing.

[2] In this work the language aspects are considered to be integrated with the function of thinking.

unintentional attention). At the same time, perceptions are transformed into the needs by the free attention and the decision process is formulated [5].

Thinking is accountable for solving our problems. An important role also is played here by the memory, which keeps the images of different objects (impressions, perceptions) at all the stages of the cognitive processes. Conscious reaction affecting the environment can be treated as a final product of thinking performed by the considered entity [6].

All the system data (representing pertinent signals) are handled by the memory read and write processes. Various memory units can be distinguished:

- sensory memory, latching and keeping stimuli
- short-lived memory is a data exchanger for the attention and thinking processes that can contain only a limited (seven) number of abstract items
- long-lived memory, having a long record of data.

Control signals are means of direct coupling between particular cognitive processes. By this means the thinking entity, for instance, is able to pay its attention to one of the discoveries made by the perception process, i.e. the attention process can ordain a deeper analysis of the object.

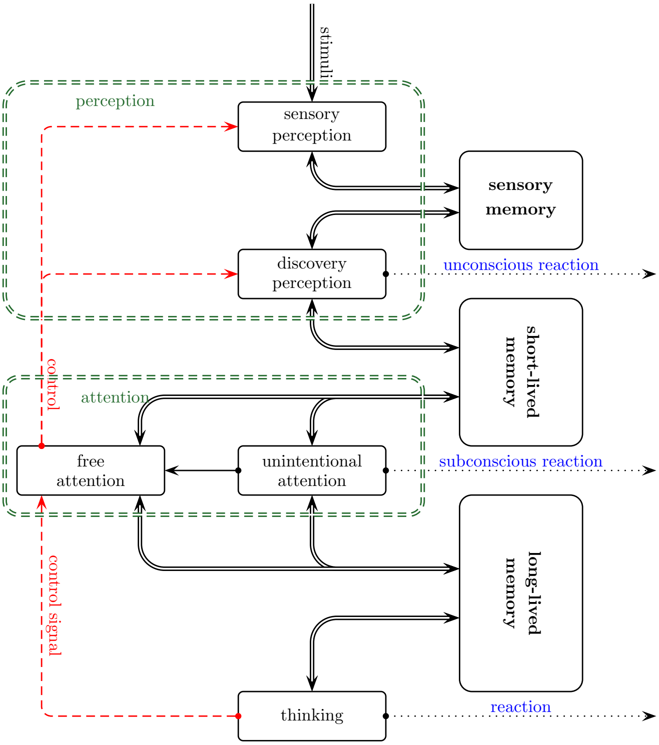


Fig. 1. Basic model of the cognitive processes

2.2 Needs in Modeling Personality Psychology

A need is an abstract state of the thinking entity experiencing a sense of dissatisfaction [7]. The stronger is the sense the harder it should be eliminated. There are a number of needs, which can be divided into several classes shown in Fig. 2.

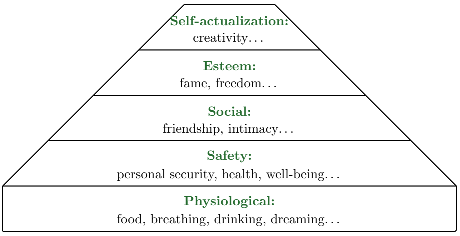


Fig. 2. Maslow's hierarchy of needs

The perspective autonomous robots should have certain additional "needs" connected to their subordinate tasks (within the assumed dependency structure), as compared to the entities created based solely on the human prototype system of needs. Soldiers obeying to army orders constitute a practical example. Such orders are directly related to the safety needs and to some extend to the "social/belonging" needs. Nevertheless, for the sake of simplicity, we assume that the external/subordinate tasks adhere to the category of safety.

3 Model Transformation

The basic model is very complicated and have too many unknown feedbacks, which have to be simplified. The more so as there are no signals of the needs. According to the literature [8] the needs are taking shape right after a preliminary selection of perceptions, *de facto* in the moment of determined by the attention. The needs, in turn, exert their influence on decision process (and, consequently, on the system reaction).

3.1 Modeling the Needs

Fuzzy set methodology will be used for the signal modeling purposes. Let us first assume that each need can be quantified (in terms of fulfillment) based on three fuzzy sets attributed to it. Their membership functions are determined by certain rudimentary parameters describing this need. Consequently, it results that each need can be in a state of satisfaction (appeasement), pre-alarm, and/or alarm. The latter absolutely requires some reaction.

A static importance of the needs can be partly determined from the pyramid of Fig. 2. Physiological needs are more important than those of safety, which, in turn, are prior to the socially belonging needs, etc.

This is not all, especially for the problem sorting point of view. Clearly, the needs should also be dynamically differentiated in terms of importance within each class (the pyramid level). The importance of the needs can be related to the degree of their fulfillment with the use of the following weighting function:

$$\omega\left(\eta_i; c\right) = \frac{1}{1 + exp\left[-\left(0.1 + 0.00025\left(c - 50\right)^2\right)\left(\left|\eta_i\right| - c\right)\right]} \tag{1}$$

where η_i represents an actual value of the degree of fulfillment of a given i-th need, and the coefficient c is a mean value of the membership parameters describing the fuzzy sets of satisfaction and alarm. An exemplary classification of such a need is shown in Fig. 3.

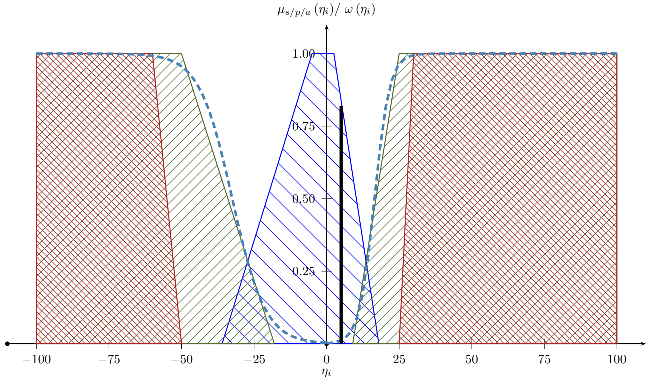


Fig. 3. Exemplary membership of an i-th need and its estimates. The bold dashed line denotes the weighting function ($\omega\left(\eta_i\right)$), the (blue) sparsely hatched-backslashed area describes the satisfaction set ($\mu_s\left(\eta_i\right)$), the (red) densely hatched-crossed area portrays the alarm set ($\mu_a\left(\eta_i\right)$), and the (green) densely hatched-slashed area means the pre-alarm set ($\mu_p\left(\eta_i\right)$). The thick vertical line marks the actual value of the fulfillment degree (η_i).

It is clear that in the moments of satisfaction the need is not important, whereas in its alarm cases the need gets the highest rank (weight).

3.2 Integral Model

In the resulting integral model shown in Fig. 4, apart from the signals of needs, there are certain emotions marked, which influence the process of recoding the impressions, and the need membership parameters, too. Taking into account the prospective robotic applications, most of the feedback signals have been replace with interactions with the environment of the entity. An exclusion has been made for the subconscious learning, which converts a frequent execution of the same activity into a strong trace in the memory [9].

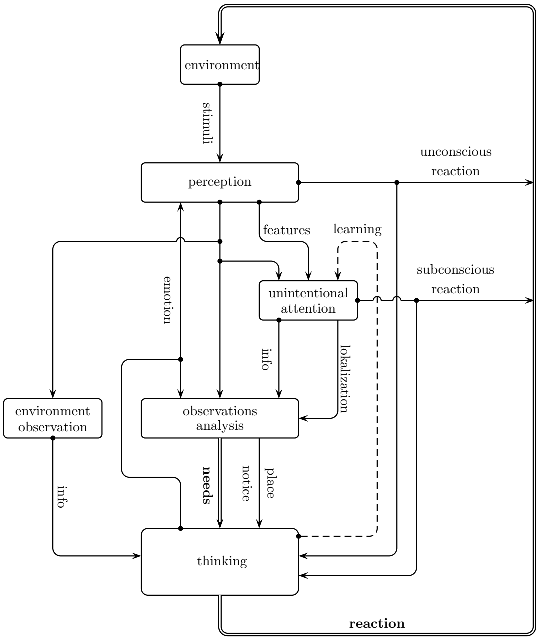


Fig. 4. Integral model of the cognitive processes with the elements of motivation

3.3 Mechanism of Selecting Reactions – A Fuzzy-Neural Network

The latter to be discussed is the thinking process. One straightforward solution implemented at this stage takes into account only a reproductive model, where all possible reactions are assumed to be known, and the only task is to select a suitable reaction according to the state of needs. Even this apparently simple task complicates when one has to consider several tens of the needs. Therefore, based *a priory* information about incremental effects on the needs, for each reaction a simulation run is performed in order to estimate its influence on the system of needs. The obtained new fulfillment degrees and importance weights of all the simulated needs are then taken into account as the input information **u** applied to the neural network of Fig. 5 (confer the symbols of Fig. 3).

The first neuron reflects the fuzzy operations between the membership of the need to the satisfaction set and the weight of the need. The second neuron considers the pre-alarm set and the third the alarm set (both use the fuzzy estimates and the need weight). The neuron of type AND is described by the following function:

$$y = f_{and}(\mathbf{u}) = \bigotimes_{i=1}^{N} (w_i \odot u_i) \tag{2}$$

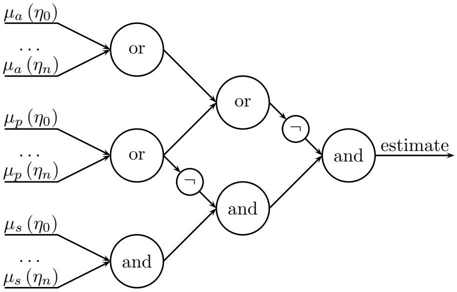


Fig. 5. Fuzzy-neural network estimating the suitability of reactions based on the actual state of needs and the simulated effects of reactions. The weights of the first layer are: $\omega(\eta_i)$; for the second layer: [[0.8 0.2] [0.1 0.9]]; and for the third layer [0.5 0.5].

where $\bigotimes$ denotes the N-argument T-norm, and $\odot$ represent the two-argument S-norm. The neuron of type OR is described as:

$$y = f_{or}(\mathbf{u}) = \bigodot_{i=1}^{N} (w_i \otimes u_i) \tag{3}$$

where $\otimes$ is the two-argument T-norm, and $\bigodot$ denotes the N-argument S-norm. Both the T-norm and the S-norm are implemented as the Einstein norms [10]:

$$\otimes_E(x, y) = \frac{xy}{2 - (x + y - xy)} \tag{4}$$

$$\odot_E(x, y) = \frac{x + y}{(1 + xy)} \tag{5}$$

The symbol ¬ represents a negation in the Yager sense:

$$N(x; s) = (1 - x^s)^{\frac{1}{s}} \tag{6}$$

In the fuzzy-neural network, estimating the effects of reactions, the negation parameter is $s = 2$. The network generates its estimation as a value from the interval $< 0, 1 >$, what results in easily sorting of reactions. The best reaction is delivered for execution, with the expectation for the improvement in the system states of needs.

4 Experiments

Let us assume that the system recognizes the following five needs: *breathing*, *health*, *friendship*, *fame* and *creativity*, and knows four types of reaction, having defined/estimated influence on its needs. The only need being in the alarm state is *fame*, while the rest of needs are *satisfied*. In such a state the system aims to satisfy the *fame* (making worse the other needs, at the same time). With reference to Tab. 1, the best reaction is *action*_3 (highest estimated mark), but also *action*_0 making better *breathing* and *health* (which are clearly more important).

Table 1. Reactions, their simulated effect on the needs and their estimation gained by the fuzzy-neural network

Reaction	breathing	health	friendship	fame	creativity	estimated mark
*action*_0	+30	+10	-5	-5	-5	0.609
*action*_1	-30	+10	-5	-5	+10	0.318
*action*_2	-30	-10	+5	+5	+5	0.323
*action*_3	+5	+10	-5	+25	-5	0.647

5 Conclusions

The paper reports the obtained results of transforming a human psychological model into an integral model of an interactive cognitive-behavioral, emotionally-driven system of making decisions conditioned both by the environment and the actual extend of the fulfillment of definite needs. A method of estimating known reactions has been proposed. The future work is devoted to the inclusion of *emotions* and *mood*. Creating an efficient generator of new reactions is also a challenging task for future research.

References

1. IEEE: IBM Unveils a New Brain Simulator (2009), http://spectrum.ieee.org/computing
2. Wells, A., Matthews, G.: Attention and Emotion: a Clinical Perspective. Erlbaum, New York (1994)
3. Maruszewski, T.: Psychology of Cognition. Gdansk Psychology Publ., Gdansk (2001) (in Polish)
4. Hebb, D.O.: Handbook of Psychology. PWN, Warsaw (1969)
5. Lewicki, A.: Cognitive Processes and Environmental Orientation. PWN, Warsaw (1960) (in Polish)
6. Monsell, S.: Control of mental processes. Unsolved Mysteries of the Mind. Tutorial Essays in Cognition, 93–148 (1996)
7. Maslow, A.H.: Toward a Psychology of Being, 2nd edn. Van Nostrand, New York (1868)
8. Necka, E., Orzechowski, J., Szymura, B.: Cognitive Psychology. PWN, Warsaw (2006) (in Polish)
9. Anderson, J.R.: Language, Memory and Thought. Erlbaum, Hillsdale (1976)
10. Leski, J.: Neuro-Fuzzy Systems. WNT, Warsaw (2008) (in Polish)
11. Kahneman, D.: Attention and Effort. Prentice Hall, Englewood Cliffs (1973)
12. Young, P.T.: Appetite, palatability and feeding habit: a critical review. Psychol. Bull. 45, 289–320 (1945)
13. Bar, M.: The proactive brain: memory for predictions. Philosophical Transactions of the Royal Society B (Theme: Predictions in the brain: Using our past to generate a future) 346, 1235–1243 (2009)
14. Davies, D.N.: Agents, emergence, emotion and representation. In: 26th Annual Conference of the IEEE, vol.4 (2000)

The Influence of Censoring for the Performance of Survival Tree Ensemble

Małgorzata Krętowska

Faculty of Computer Science
Bialystok University of Technology
Wiejska 45a, 15-351 Białystok, Poland
m.kretowska@pb.edu.pl

Abstract. One of the main objectives in survival analysis is prediction the time of failure occurrence. It is done on a base of learning sets, which contain incomplete (censored) information on patients failure times. Proposed predictors should allow to cope with censored data. In the paper the influence of censoring for the performance of dipolar tree ensemble was investigated. The prediction ability of the model was verified by several measures, such as direct and indirect estimators of absolute predictive errors: $\tilde{D}_{S,x}$, $\hat{D}_x$ and explained variation. The analysis is conducted on the base of artificial data, generated with different values of censoring rate.

1 Introduction

Censoring is a term used in survival analysis. It describes the data with incomplete information about failure time. In survival data, each patient is characterized by the feature vector $\mathbf{x}$ and corresponding survival time t, which is counted from the beginning event (e.g. surgery). If we define the event of interest (e.g death, disease relapse) and the cut-off date (how long the patients will be under investigation), time t has two different meanings:

- *failure time* - if the observation is finished with the event of interest,
- *follow-up time* - if the patient is observed to the cut-off date or the observation is finished by another event not connected directly with the aim of medical experiment

Observations, for which only the follow-up time is given are called *censored*. They do not contain the exact knowledge of the failure time. We only know, how long the patient was observed. To distinguish between the failure and the follow-up time, a new binary variable is introduced - *failure indicator* δ, which is equal to 0 for censored cases, and 1 otherwise.

The presence of censored cases causes some problems in analysis of survival data. Because the percentage of censored observations may be high (e.g. Melanoma Malignum dataset [1] - 72%, Primary Biliary Cirrhosis dataset [5] - 60% of censored observations), they should not be ignored. There exist a number

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 524–531, 2010.
© Springer-Verlag Berlin Heidelberg 2010

of statistical methods which were developed to analyze censored data (e.g. Cox's proportional hazards model [3]), but they require some additional assumptions to fulfill. If the requirements are difficult to obey, other non-statistical techniques are proposed. Here, neural networks and regression trees are the most common ones. Such methods do not always are adapted directly to censoring data. In many cases the authors use standard tools with omitting the majority of censored cases or with earlier estimation of the failure time for censored cases, so called *imputation*. Lapuerta *et al.* [11] proposed an additional neural network structure for failure time estimation, the use of Cox's regression model is introduced by De Laurentiis and Ravdin [4] and Ripley [12]. Ignoring the censored cases as well as the imputation process allow receiving the biased results. In the first case, only the part of information is used, in the other, new information is added to the data.

In the paper the influence of censoring on the performance of dipolar based ensemble is investigated. The dipolar tree ensemble [8] as well as the neural network ensemble [9], use censored cases during the learning process, while creating the dipolar criterion function. Because better prediction ability was received for dipolar tree ensemble [10], this model is taken into account. The analysis is conducted on the base of artificial datasets, generated with different values of censoring rate ($0 - 70\%$). Predictive ability of the models is evaluated using the indirect an direct estimators of absolute predictive errors and explained variation ([14,13]).

The paper is organized as follows. Section 2 describes the distribution functions of failure time and introduces the idea of Kaplan-Meier survival function. In Section 3 the idea of dipoles and dipolar tree ensemble is presented. Measures of predictive ability for censored data are described in Section 4. Experimental results are presented in Section 5. Section 6 summarizes the results.

2 Distribution Functions of Survival Time

Let T^0 denotes the true survival time and C denotes the true censoring time with distribution functions F and G respectively. We observe random variable $O = (T, \Delta, \mathbf{X})$, where $T = \min(T^0, C)$ is the time to event, $\Delta = I(T \leq C)$ is a censoring indicator and $\mathbf{X} = (X_1, ..., X_N)$ denotes the set of N covariates from a sample space χ. We have learning sample $L = (\mathbf{x}_i, t_i, \delta_i)$, $i = 1, 2, ..., n$, where $\mathbf{x}_i$ is N-dimensional covariates vector, t_i - survival time and δ_i - failure indicator, which is equal to 0 for censored cases and 1 for uncensored ones.

The distribution of random variable T may be described by several functions:

- survival function
$$S(t) = P(T > t) \tag{1}$$
where $P(\bullet)$ means probability, $S(0) = 1$ and $\lim_{t\to\infty} S(t) = 0$
- density function
$$f(t) = \lim_{\triangle t \to 0} \frac{P(t \leq T < t + \triangle t)}{\triangle t} \tag{2}$$

where $f(t)dt$ is the unconditional probability of failure in the infinitesimal interval $(t, t + dt)$.

- hazard function

$$\lambda(t) = \lim_{\triangle t \to 0} \frac{P(t \leq T < t + \triangle t | T \geq t)}{\triangle t} \tag{3}$$

where $\lambda(t)dt$ is the probability of failure in the in infinitesimal interval $(t, t + dt)$, given survival at time t.

The estimation of survival function $S(t)$ may be done by using the Kaplan-Meier product limit estimator [6], which is calculated on the base of learning sample L and is denoted by $\hat{S}(t)$:

$$\hat{S}(t) = \prod_{j|t_{(j)} \leq t} \left(\frac{m_j - d_j}{m_j} \right) \tag{4}$$

where $t_{(1)} < t_{(2)} < \ldots < t_{(D)}$ are distinct, ordered survival times from the learning sample L, in which the event of interest occurred, d_j is the number of events at time $t_{(j)}$ and m_j is the number of patients at risk at $t_{(j)}$ (i.e., the number of patients who are alive at $t_{(j)}$ or experience the event of interest at $t_{(j)}$).

The 'patients specific' survival probability function is given by $S(t|\mathbf{x}) = P(T > t|\mathbf{X} = \mathbf{x})$. The conditional survival probability function for the new patient with covariates vector $\mathbf{x}_{new}$ is denoted by $\hat{S}(t|\mathbf{x}_{new})$.

3 Survival Tree Ensemble

Individual survival tree being a part of the complex predictor [8] is a kind of binary regression tree. Each internal node contains a split, which tests the value of an expression of the covariates. In the proposed approach the split is equivalent to the hyper-plane $H(\mathbf{w}, \theta) = \{(\mathbf{w}, \mathbf{x}) :< \mathbf{w}, \mathbf{x} >= \theta\}$.

Establishing the structure of the tree (the number of internal nodes) and the values of hyper-planes parameters $(\mathbf{w}, \theta)$ are based on the concept of dipoles [2]. The dipole is a pair of different covariate vectors $(\mathbf{x}_i, \mathbf{x}_j)$ from the learning set. Mixed and pure dipoles are distinguished. Assuming that the analysis aims at dividing the feature space into such areas, which would include the patients with similar survival times, pure dipoles are created between pairs of feature vectors, for which the difference of failure times is small, mixed dipoles - between pairs with distant failure times. Taking into account censored cases the following rules of dipole construction can be formulated:

1. a pair of feature vectors $(\mathbf{x}_i, \mathbf{x}_j)$ forms the pure dipole, if
   - $\delta_i = \delta_j = 1$ and $|t_i - t_j| < \eta$
2. a pair of feature vectors $(\mathbf{x}_i, \mathbf{x}_j)$ forms the mixed dipole, if
   - $\delta_i = \delta_j = 1$ and $|t_i - t_j| > \zeta$
   - $(\delta_i = 0, \delta_j = 1$ and $t_i - t_j > \zeta)$ or $(\delta_i = 1, \delta_j = 0$ and $t_j - t_i > \zeta)$

Parameters η and ζ are equal to quartiles of absolute values of differences between uncensored survival times. Basing on the earlier experiments, the parameter η is fixed as 0.2 quartile and ζ - 0.6. An example of dipoles construction by censored and uncensored observation is presented in Fig. 1

The increasing number of censored cases may decrease the number of pure dipoles as well as the mixed ones.

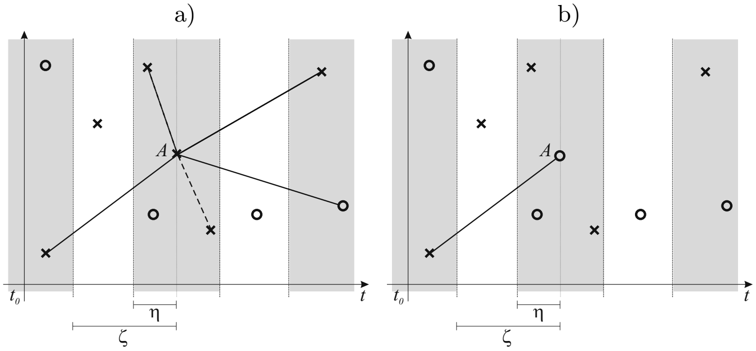


Fig. 1. Construction of pure (solid line) and mixed (dotted line) dipoles by a) uncensored observation; b) censored observation

The hyper-planes $H(\mathbf{w}, \theta)$ in the internal nodes of a tree are calculated by minimization of dipolar criterion function (detailed description may be fond in [8]). This is equivalent with division of possibly high number of mixed dipoles and possibly low number of pure ones constructed for a given dataset. The tree induction algorithm starts from the root, so in the root node, the dipolar criterion function is calculated on the base of dipoles created for the whole learning set. The dipolar criterion function for consecutive nodes of a tree are designed on the base on those feature vectors that reached the node. The induction of survival tree is stopped if one of the following conditions is fulfilled: 1) all the mixed dipoles are divided; 2) the set that reach the node consists less than 5 uncensored cases.

The survival tree ensemble algorithm leading to receive the aggregated survival function $\hat{S}(t|\mathbf{x}_n)$ is as follows:

1. Draw k bootstrap samples $(L_1, L_2, \ldots, L_k)$ of size n with replacement from L
2. Induction of dipolar survival tree $T(L_i)$ based on each bootstrap sample L_i
3. For each tree $T(L_i)$, distinguish the set of observations $L_i(\mathbf{x}_n)$ which belongs to the same terminal node as $\mathbf{x}_n$
4. Build aggregated sample $L_A(\mathbf{x}_n) = [L_1(\mathbf{x}_n), L_2(\mathbf{x}_n), \ldots, L_k(\mathbf{x}_n)]$
5. Compute the Kaplan-Meier aggregated survival function for a new observation $\mathbf{x}_n$ as $\hat{S}_A(t|\mathbf{x}_n)$.

The predicted value of exact failure time for observation $\mathbf{x}_n$ may be calculated as the median value of $\hat{S}_A(t|\mathbf{x}_n)$.

4 Evaluation of Predictive Ability

Predictive ability of the model is calculated using the measures adapting for censoring. One of them is direct estimator of absolute predictive error (APE) [14], calculated for each distinct failure time $t_{(j)}$:

$$\begin{aligned}\hat{M}(t_{(j)}) = \quad & \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \Big[I(t_i > t_{(j)})(1-\hat{S}(t_{(j)})) + \delta_i I(t_i \leq t_{(j)})\hat{S}(t_{(j)}) + \\ & (1-\delta_i)I(t_i \leq t_{(j)}) \Big\{ (1-\hat{S}(t_{(j)}))\tfrac{\hat{S}(t_{(j)})}{\hat{S}(t_i)} + \hat{S}(t_{(j)})(1-\tfrac{\hat{S}(t_{(j)})}{\hat{S}(t_i)}) \Big\} \Big]\end{aligned} \quad (5)$$

where $I(condition)$ is equal to 1 if the condition is fulfilled and 0 otherwise. The measure with covariates ($\hat{M}(t_{(j)}|\mathbf{x})$) is obtained by replacing $\hat{S}(t_{(j)})$ by $\hat{S}(t_{(j)}|\mathbf{x})$ and $\hat{S}(t_i)$ by $\hat{S}(t_i|\mathbf{x})$. To receive overall estimators of APE with ($\hat{D}_x$) and without covariates ($\hat{D}$) the weighed averages of estimators over failure times are calculated:

$$\hat{D} = w^{-1} \sum_j \hat{G}(t_{(j)})^{-1} d_j \hat{M}(t_{(j)}) \quad (6)$$

$$\hat{D}_{\mathbf{x}} = w^{-1} \sum_j \hat{G}(t_{(j)})^{-1} d_j \hat{M}(t_{(j)}|\mathbf{x}) \quad (7)$$

where $w = \sum_j \hat{G}(t_{(j)})^{-1} d_j$, d_j is the number of events at time $t_{(j)}$ and $\hat{G}(t)$ denotes the Kaplan-Meier estimator of the censoring distribution. It is calculated on the base of observations $(t_i, 1-\delta_i)$.

The indirect estimation of predictive accuracy was proposed by Schemper [13]. In the approach the estimates (without $\tilde{M}(t_{(j)})$ and with covariates $\tilde{M}(t_{(j)}|\mathbf{x})$) are defined by

$$\tilde{M}(t_{(j)}) = 2\hat{S}(t_{(j)})(1-\hat{S}(t_{(j)})) \quad (8)$$

$$\tilde{M}(t_{(j)}|\mathbf{x}) = 2n^{-1} \sum_i \hat{S}(t_{(j)}|\mathbf{x}_i)(1-\hat{S}(t_{(j)})|\mathbf{x}_i) \quad (9)$$

The overall estimators of predictive accuracy with ($\tilde{D}_{S,\mathbf{x}}$) and without ($\tilde{D}_S$) covariates are calculated similarly to the estimators $\hat{D}_{\mathbf{x}}$ and $\hat{D}$. The only change is replacing $\hat{M}(t_{(j)})$ and $\hat{M}(t_{(j)}|\mathbf{x})$ by $\tilde{M}(t_{(j)})$ and $\tilde{M}(t_{(j)}|\mathbf{x})$ respectively.

Based on the above overall estimators of absolute predictive error, explained variation (EV) can be defined as $\tilde{V}_S = \frac{\tilde{D}_S - \tilde{D}_{S,\mathbf{x}}}{\tilde{D}_S}$ and $\hat{V} = \frac{\hat{D} - \hat{D}_{\mathbf{x}}}{\hat{D}}$.

5 Experimental Results

The influence of censoring for the performance of survival tree ensemble was evaluated on the base of several simulated datasets. An exponential survival distribution was assumed for the proportional hazards models. Let

$$\lambda(t, \mathbf{x}) = \exp\left\{ \sum_{i=1}^{N} \beta_i(t)x_i + \sum\sum_{i \neq j} \gamma_{ij}(t)x_i x_j + \sum_{i \neq j \neq k} \gamma_{ijk}(t)x_i x_j x_k \right\} \quad (10)$$

be the hazard at any time t given N covariates $\mathbf{x}$, the survival times were then generated using inverse probability transformations [15]. The following datasets were considered:

- (a) $N = 2$, with $\beta_1 = 0.25$, $\beta_2 = 0.5$; x_1 and x_2 have normal distribution $N(0,1)$
- (b) the same as in (a), except that an interaction $\gamma_{12} = 0.2$ was assumed
- (c) $N = 4$, with $\beta_1 = 1$, $\beta_2 = 0.25$, $\beta_3 = 1$, $\beta_4 = 0.5$; x_1, x_2 and x_3 have normal distribution $N(0,1)$ and x_4 have Bernoulli distribution with $n = 1$ and $p = 0.5$.
- (d) the same as in (c), except that interactions $\gamma_{12} = \gamma_{23} = 0.2$ and $\gamma_{123} = \gamma_{234} = 0.5$ were assumed

Each dataset was generated with 0, 10, 20, 30, 40, 50, 60 and 70 per cent of censoring, number of cases $n = 400$. Censoring time was exponentially distributed and independent of the survival time.

All the experiments were performed using the ensemble of 100 survival trees ST. The measures of predictive accuracy were calculated on the base of learning sample L. To calculate the aggregated survival function for a given example $\mathbf{x}$ from learning set L, only such ST_i $(i = 1, 2, \ldots, 100)$ were taken into consideration, for which $\mathbf{x}$ was not belonged to learning set L_i (i.e. $\mathbf{x}$ did not participate in the learning process of the ST_i). On average, each learning set L_i do not contain 1/3 elements from L [7].

In figure 2 the values of described earlier predictive measures for the generated datasets are presented. In each case values of direct and indirect APE without covariates were the same ($dAPE = iAPE$), so in the figure only the indirect measure is shown. The influence of censoring rate for the predictive accuracy is similar for each data. As we can see, relatively small values of censoring rate do not influence much the predictive ability of the model. For datasets without interactions ((a), (b)) we can observe similar values of explained variation for data with censoring rate less then 30%. For dataset (a) the values of indirect (direct) explained variance is equal to 0.95(0.97), 0.9(0.91), 0.93(0.94), respectively for 0, 10 and 20 censoring rate, for dataset (c) we receive 0.93(0.96), 0.89(0.9), 0.93(0.96), respectively. For higher values of censoring rate the predictive ability of the model is poor. The value of absolute predictive error increases and hence explained variation is getting close to 0.

Similar behavior is observed for the datasets with interactions. The predictive ability of received models for the censoring rates between 0 and 30% is also high and equal to 0.85(0.89), 0.82(0.85), 0.8(0.83) and 0,84(0.87) for dataset (b) and 0.73(0.78), 0.74(0.78), 0.75(0.78), 0.76(0.8) for dataset (d). The increasing value of censoring rate decreases the predictive ability of the model.

Analyzing the results received for the models build for data with high percentage of censored cases, it was noticed that for many observations the predicted failure times were undefined. The failure time for a given patient $\mathbf{x}_{new}$ is calculated as the median value of the received aggregated Kaplan-Meier survival function. If the number of censored cases is high, the values of $\hat{S}(t|\mathbf{x}_{new})$ are above 0.5, so the median value can not be calculated.

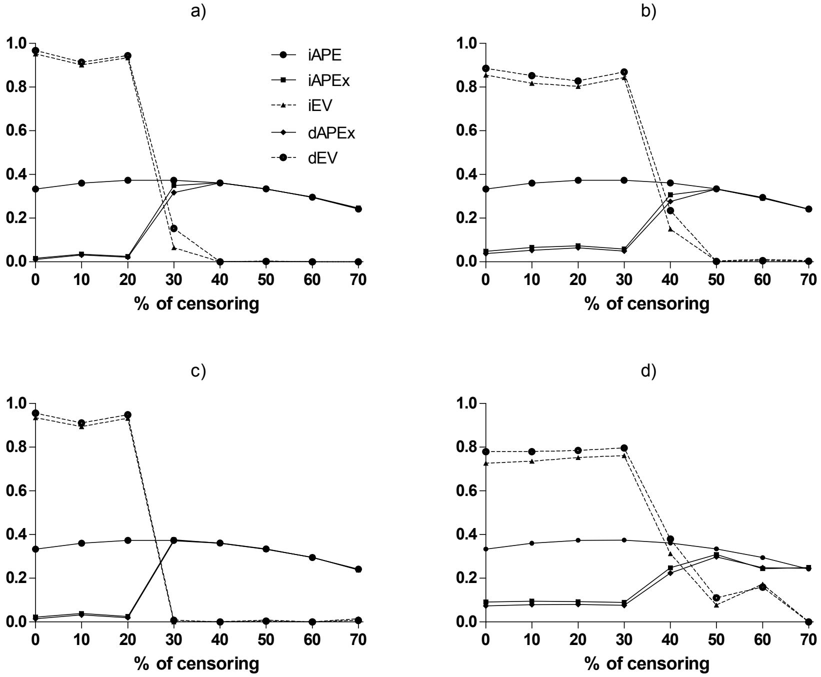


Fig. 2. Predictive measures for generated datasets with different value of censoring rate

6 Conclusions

In the paper the influence of censoring for the performance of dipolar tree ensemble was investigated. The prediction ability of the models was verified by several measures, such as direct and indirect estimators of absolute predictive errors: $\tilde{D}_{S,x}$, $\hat{D}_x$ and explained variation. The results received for artificial datasets, generated with different number of censoring rates (0% – 70%), suggest that the number of inputs or the presence of interaction do not influence substantially the performance of the predictor. Dipolar tree ensembles generated for the datasets with relatively small number of censored cases (less then 30% or 40% per cent, for datasets without and with interactions respectively) have very good predictive ability. For higher values of censoring rate the explained variation was decreasing. Because the high number of censored cases makes the exact prediction of the failure time impossible, the patient survival should be described by the aggregated Kaplan-Meier survival function.

Acknowledgements. This work was supported by the grant W/WI/4/08 from Bialystok Technical University.

References

1. Andersen, P.K., Borgan, O., Gill, R.D.: Statistical Models based on Counting Processes. Springer, New York (1993)
2. Bobrowski, L., Krętowska, M., Krętowski, M.: Design of neural classifying networks by using dipolar criterions. In: Proc. of the Third Conference on Neural Networks and Their Applications, Kule, Poland, pp. 689–694 (1997)
3. Cox, D.R.: Regression models and life tables (with discussion). Journal of the Royal Statistical Society B 34, 187–220 (1972)
4. De Laurentiis, M., Ravdin, P.M.: Survival analysis of censored data: neural network analysis detection of complex interactions between variables. Breast Cancer Research and Treatment 32(1), 113–118 (1994)
5. Fleming, T.R., Harrington, D.P.: Counting Processes and Survival Analysis. John Wiley & Sons, Inc, Chichester (1991)
6. Kaplan, E.L., Meier, P.: Nonparametric estimation from incomplete observations. Journal of the American Statistical Association 5, 457–481 (1958)
7. Koronacki, J., Cwik, J.: Statistical learning systems. Wydawnictwa Naukowo-Techniczne, Warsaw (2005) (in Polish)
8. Krętowska, M.: Random forest of dipolar trees for survival prediction. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 909–918. Springer, Heidelberg (2006)
9. Krętowska, M.: Ensemble of dipolar neural networks in application to survival data. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 78–88. Springer, Heidelberg (2008)
10. Krętowska, M.: Prognostic abilities of dipoles based ensembles comparative analysis. Zeszyty Naukowe Politechniki Bialostockiej. Informatyka 4, 73–83 (2009)
11. Lapuerta, P., Azen, S.P., LaBree, L.: Use of neural networks in predicting the risk of coronary artery disease. Computers and Biomedical Research 28, 38–52 (1995)
12. Ripley, R.M.: Neural networks for breast cancer prognosis. PhD thesis, Department of Engineering Science, University of Oxford (1998)
13. Schemper, M.: Predictive accuracy and explained variation. Statistics in Medicine 22, 2299–2308 (2003)
14. Schemper, M., Henderson, R.: Predictive accuracy and explained variation in Cox regression. Biometrics 56, 249–255 (2000)
15. Xiang, A., Lapuerta, P., Ryutov, A., Buckley, J., Azen, S.: Comparison of the performance of neural network methods and Cox regression for censored survival data. Computational Statistics & Data Analysis 34, 243–257 (2000)

Clustering Polish Texts with Latent Semantic Analysis

Marcin Kuta and Jacek Kitowski

Institute of Computer Science, AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
{mkuta,kito}@agh.edu.pl

Abstract. The document clustering is an important technique of Natural Language Processing (NLP). The paper presents performance of partitional and agglomerative algorithms applied to clustering large number of Polish newspaper articles. We investigate different representations of the documents. The focus of the paper is on the applicability of the Latent Semantic Analysis to such clustering for Polish.

Keywords: document clustering, latent semantic analysis, part-of-speech tagging, natural language processing.

1 Introduction

The document clustering is an important technique applied to unsupervised organization of documents or web search results, creating semantic nets (e.g. wordnets) [1], information retrieval and topic extraction. Unfortunately, few works have been done for Polish and research focused on impact of stemming or stopwords on quality of web search results.

The aim of the paper is to investigate efficiency of Vector Space Model (VSM) and Latent Semantic Analysis (LSA) when applied to clustering large number of articles in the Polish language, extracted from one of the main Polish newspapers. We examine both partitional and agglomerative algorithms, several cluster criterion functions used for optimization of cluster bisection or merging process. For LSA model we analyse the number of dimensions which is the most suitable for performing clustering. Within VSM and LSA models we compare *tfidf* and *logent* term weighting schemes and representation of the corpus with different feature selection schemes.

2 Corpus Preparation

The clustering experiments have been conducted with the corpus of the Polish daily *Rzeczpospolita* [2] (henceforth the ROL corpus). The whole corpus consists of over 190,000 articles published in years 1993–2002. Each document is originally stored in the html format and annotated by editorial board with information about category and other metadata containing author, issue date, keywords, etc.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 532–539, 2010.
© Springer-Verlag Berlin Heidelberg 2010

For the purpose of clustering experiments we selected a subset of 10,000 articles, belonging to 6 general, disjoint categories: culture, economy, law, national news, international news, and sport.

In order to conduct experiments we performed several transformations of documents. The preliminary processing consists from the following steps: transformation to pure UTF8 text format, encoding enhancements of some non alphanumeric characters, and splitting the uniform text stream into sentences.

The feature reduction process (lemmatisation) and feature selection based on part-of-speech (POS) of a term requires POS tagging in an auxiliary phase. The corpus has been tagged with the TnT tagger [3], trained previously on the modified Frequency Dictionary of Contemporary Polish corpus (the m-FDCP corpus) and thus provided with the model of tagging for Polish [4]. The TnT tagger achieves high tagging accuracy for Polish and is able to cope with very long sentences [5].

Next the proper lemmatisation phase is conducted with use of the morphological analyser of Polish – Morfeusz [6]. The format of the tagset required by Morfeusz as a hint slightly differs from the format of the tags emitted by TnT – necessary translation from the TnT to Morfeusz specification have been performed. Finally, terms bearing no information, contained within stop list are filtered out.

We consider four representations of the corpus, which we call respectively base, noun, bigram and trigram model. The base representation of the corpus contains only terms with open-class POS, i.e., nouns, verbs, adjectives and adverbs. In the noun representation only noun terms are present. The bigram representation extends the base model with bigrams which occurred more than 6 times in the entire corpus. The trigram representation adds to the bigram model trigrams occurring more than 9 times in the entire corpus. We took into account only bigrams and trigrams composed from nouns, adjectives, verbs or adverbs (possibly intermixed).

VSM represents each text document as a vector of terms, where each dimension corresponds to a separate term. A term present in a document is assigned a value (weight) in a vector. We reflected two most important weighting schemes, tfidf and logent, defined according to below equations:

$$tfidf = f_{ij} \cdot \frac{N}{df_j}, \tag{1}$$

$$logent = f_{ij} \cdot \left(1 + \sum_{j=1}^{N} \frac{\frac{f_{ij}}{F_j} \log(\frac{f_{ij}}{F_j})}{\log N}\right), \tag{2}$$

where N – total number of documents; df_j – number of documents containing term t_j; f_{ij} – the number of occurrences of term t_j in document d_i; F_j – total number of occurrences of term t_j in all documents.

LSA [7] is a technique of analyzing relationships between a set of documents and the terms they contain by producing a set of concepts represented in a semantic low dimensional space. It is achieved by Singular Value Decomposition (SVD) of a term-document matrix and finding its low-rank approximation.

3 Clustering Algorithms

We examined 5 clustering algorithms: repeated bisections (RB), repeated bisections with refinements (RBR), direct method, agglomerative algorithm [8] and density clustering (DC) algorithm developed at Institute of Computer Science at AGH-UST by M. Kurdziel and K. Boryczko [9].

The RB method finds cluster solution by a sequence of repeated bisections. Each bisection is chosen in a way that optimizes (maximizes or minimizes) value described by a clustering criterion function. The RBR method improves the former one by executing a global optimization at the end of the algorithm. The direct method computes solution by simultaneously finding all clusters. Above methods belong to the family of partitional algorithms. The agglomerative algorithm starts with a set of clusters, each containing exactly one document. Clusters are merged until the required number of clusters is obtained. Contrary to the previous algorithms, the criterion function does not control bisections but the merging process, i.e., performs local optimization.

The DC method exploits differences in density of clusters. This algorithm does not need criterion function.

With RBR method we considered the following criterion functions [10]:

$$\mathcal{I}_1 = \max \sum_{i=1}^{k} \frac{1}{n_i} \Big(\sum_{u,v \in S_i} sim(u,v) \Big), \qquad \mathcal{I}_2 = \max \sum_{i=1}^{k} \sqrt{\sum_{u,v \in S_i} sim(u,v)}, \tag{3a-3b}$$

$$\mathcal{E}_1 = \min \sum_{i=1}^{k} n_i \frac{\sum_{u \in S_i, v \in S} sim(u,v)}{\sqrt{\sum_{u,v \in S_i} sim(u,v)}}, \tag{3c}$$

$$\mathcal{G}_1 = \min \sum_{i=1}^{k} \frac{\sum_{u \in S_i, v \in S} sim(u,v)}{\sum_{u,v \in S_i} sim(u,v)}, \qquad \mathcal{G}_1' = \min \sum_{i=1}^{k} n_i^2 \frac{\sum_{u \in S_i, v \in S} sim(u,v)}{\sum_{u,v \in S_i} sim(u,v)}, \tag{3d-3e}$$

$$\mathcal{H}_1 = \max \frac{\mathcal{I}_1}{\mathcal{E}_1}, \qquad \mathcal{H}_2 = \max \frac{\mathcal{I}_2}{\mathcal{E}_1}, \tag{3f-3g}$$

where $sim(u,v)$ stands for similarity between documents u, v; k – the total number of clusters; S_i – i-th cluster of cardinality n_i; S – the set of all documents. As a similarity measure sim the cosine similarity have been always used. RB and direct methods have been tested with the $\mathcal{I}_2$ function.

For the agglomerative algorithm we have tested the criterion functions: $\mathcal{I}_2$, $\mathcal{E}_1$, $\mathcal{G}_1$, $\mathcal{G}_1'$, $\mathcal{H}_1$, $\mathcal{H}_2$ and functions which are specific for agglomerative algorithms, i.e., single-link (slink), complete-link (clink):

$$slink = \max_{u \in S_i, v \in S_j} sim(u,v), \qquad clink = \min_{u \in S_i, v \in S_j} sim(u,v), \tag{4a-4b}$$

and unweighted pairwise group method with averages (upgma):

$$upgma = \frac{1}{|S_i| \cdot |S_j|} \sum_{u \in S_i, v \in S_j} sim(u,v) \ . \tag{4c}$$

The clustering experiments have been conducted with the CLUTO toolkit [11]. The only exception are experiments with the DC algorithm, which could be done due to courtesy of K. Boryczko and M. Kurdziel. The SVD decomposition have been computed with the SVDLIBC library [12]. Experiments were performed at the ACC Cyfronet AGH-UST site using one processor of the SGI Altix 3700 SMP supercomputer, equipped with 256 1.5 GHz Intel Itanium 2 processors, 512 GB RAM.

4 Results

Quality of a cluster solution have been evaluated in terms of the purity measure, defined as follows:

$$Purity = \frac{1}{N} \sum_k \max_j |c_k \cap l_j| \ , \tag{5}$$

where N – number of documents, $C = \{\, c_1, \ldots, c_K \,\}$ denotes a set of found clusters by an algorithm and $L = \{\, l_1, \ldots, l_J \,\}$ is a set of golden clusters.

Results of top six clustering algorithms are presented in Table 1 and detailed results of three experiments (out of six conducted) in Figs. 1–3.

Table 1. Top six results of clustering algorithms

No	Method	Criterion function	Representation of corpus	Number of dimensions	Purity
1	RBR	$\mathcal{H}_1$	bigram	50	81.42
2	RBR	$\mathcal{H}_1$	trigram	50	81.40
3	Direct	$\mathcal{I}_2$	bigram	50	81.27
4	Direct	$\mathcal{I}_2$	trigram	50	80.79
5	RBR	$\mathcal{H}_1$	base	50	80.52
6	RBR	$\mathcal{I}_1$	base	100	80.29

Due to the usage of the random generator, each partitional method (separately for each criterion function) have been run ten times and an arithmetic mean of the purity reported as a result.

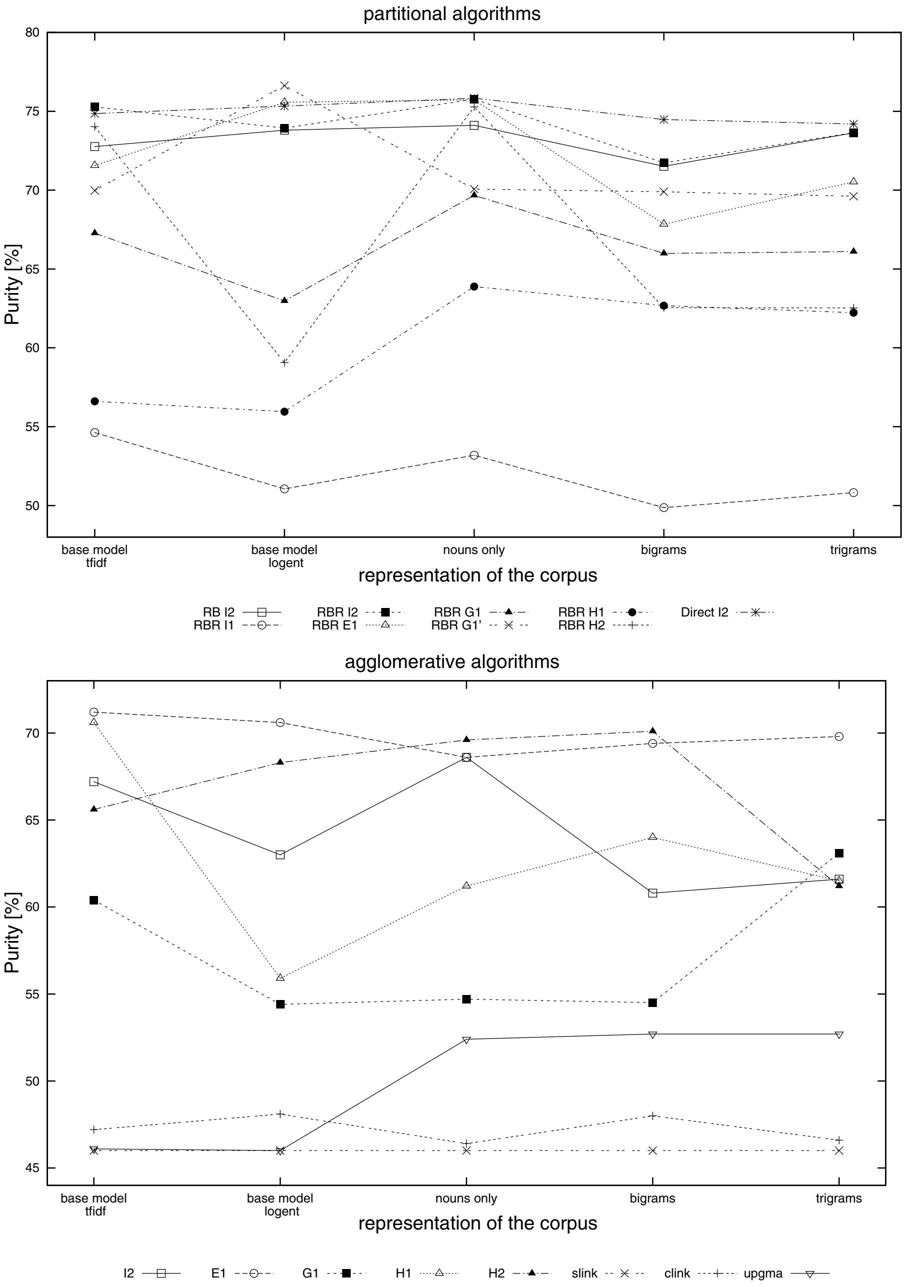


Fig. 1. Purity of the divisive and agglomerative clustering algorithms for different representations of the corpus

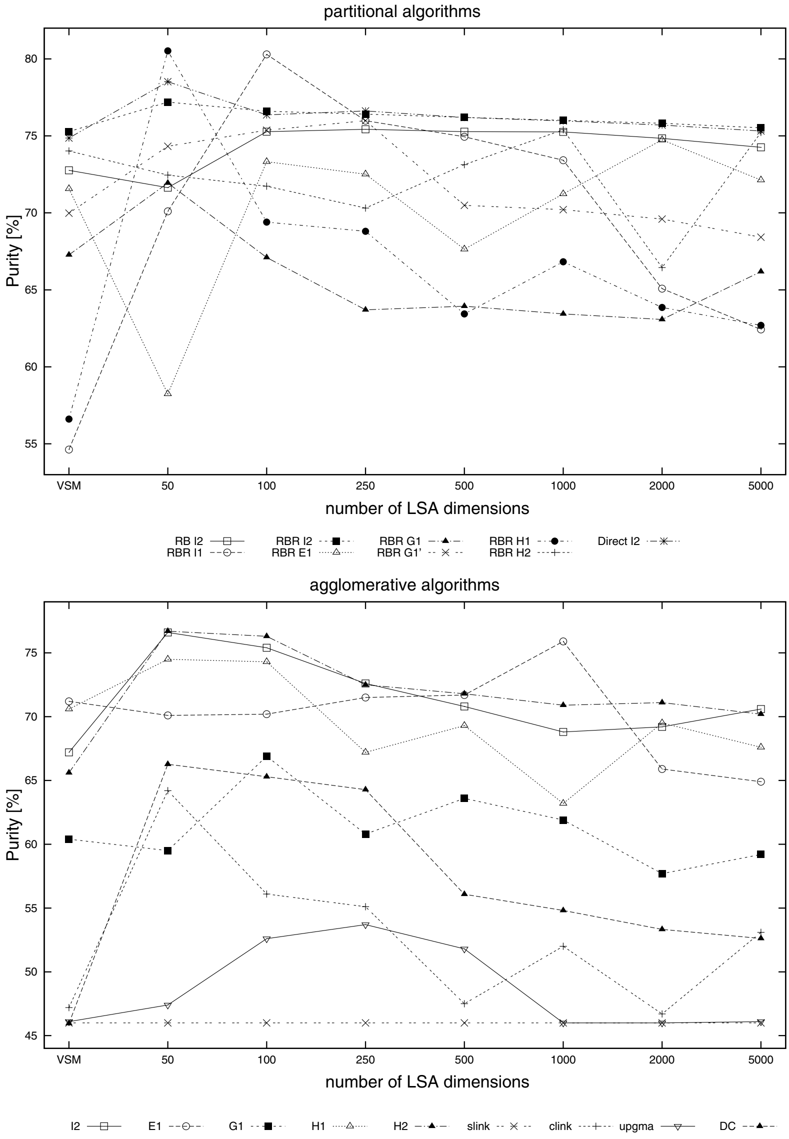


Fig. 2. Purity of the divisive and agglomerative clustering algorithms as a function of number of dimensions. Base representation of the corpus with the tfidf weighting scheme.

5 Conclusions

All experiments show that partitional algorithms achieve better results than agglomerative algorithms in clustering Polish documents when small number of clusters come into play.

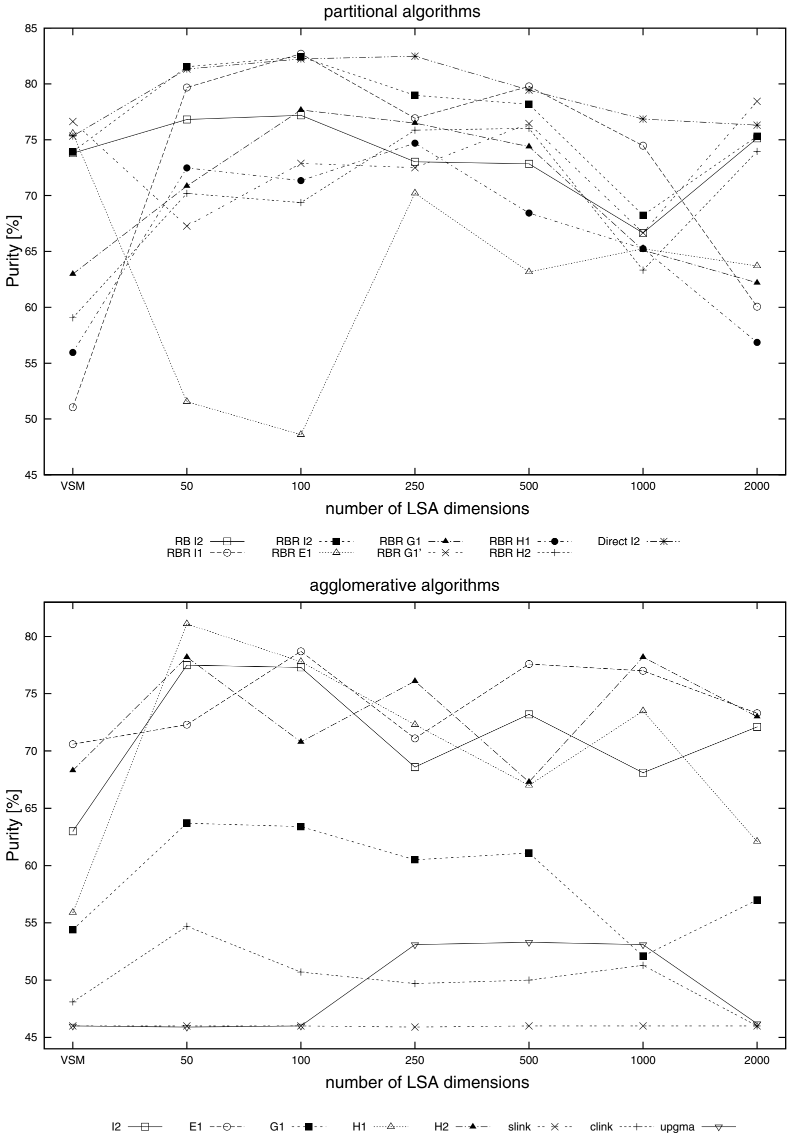


Fig. 3. Purity of the divisive and agglomerative clustering algorithms as a function of number of dimensions. Base representation of the corpus with the logent weighting scheme.

First experiment, which focuses only on the representation of the corpus by terms, shows that noun representation of the corpus is the most suitable (Fig. 1).

Subsequent experiments compared the VSM model with the LSA model for different representations of the corpus and analysed the performance of algorithms according to the number of LSA dimensions (Figs. 2, 3). The experiments show that in case of almost all algorithms LSA transformation lead to much better clustering quality than the classic VSM model. For almost all methods optimal number of dimensions equals 50 or 100 and quality of clustering decreases with further growth of number of dimensions. The highest purity is achieved by the RBR method with the $\mathcal{H}_1$ criterion function on the corpus represented with bigrams, processed with LSA transformation, and equals 81.42%.

Only results of the slink function remained insensitive to shift from the VSM to LSA model and change of the number of dimensions.

While LSA performs better than VSM in terms of the purity measure, time requirements are major shortcoming of the former algorithm.

Acknowledgments. This research is supported by the AGH University of Science and Technology (AGH-UST) grant no. 11.11.120.865. ACC CYFRONET AGH is acknowledged for the computing time. Special thanks go to Prof. K. Boryczko and Dr. M. Kurdziel from AGH-UST for making their density clustering algorithm accessible and help with its evaluation on the ROL corpus. Separate thanks go to Prof. W. Lubaszewski for his valuable remarks.

References

1. Broda, B., Piasecki, M.: Experiments in clustering documents for automatic acquisition of lexical semantic networks for Polish. In: Proc. of the 16th Int. Conf. Intelligent Information Systems, pp. 203–212 (2008)
2. Weiss, D.: The corpus of the Polish daily *Rzeczpospolita* (1993-2002), http://www.cs.put.poznan.pl/dweiss/rzeczpospolita
3. Brants, T.: TnT – a statistical part-of-speech tagger. In: Proc. of the 6th Applied Natural Language Processing Conf., pp. 224–231 (2000)
4. Kuta, M., Chrząszcz, P., Kitowski, J.: Increasing quality of the Corpus of Frequency Dictionary of Contemporary Polish for morphosyntactic tagging of the Polish language. Computing and Informatics 28(3), 319–338 (2009)
5. Kuta, M., Wójcik, W., Wrzeszcz, M., Kitowski, J.: Application of stacked methods to part-of-speech tagging of Polish. In: Proc. of the 8th Int. Conf. on Parallel Processing and Applied Mathematics, PPAM 2009 (2009)
6. Woliński, M.: Morfeusz - a practical tool for the morphological analysis of Polish. In: Proc. of the Int. Conf. Intelligent Information Systems, pp. 503–512 (2006)
7. Landauer, T., Dumais, S.: A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. Psychological Review 104(2), 211–240 (1997)
8. Zhao, Y., Karypis, G.: Hierarchical clustering algorithms for document datasets. Data Mining and Knowledge Discovery 10(2), 141–168 (2005)
9. Kurdziel, M.: Visual Clustering Methods for Pattern Recognition in Biomedical Data. PhD thesis, University of Science and Technology (2010)
10. Zhao, Y., Karypis, G.: Criterion functions for document clustering. Experiments and analysis. Technical Report 01–40, University of Minnesota, Department of Computer Science / Army HPC Research Center Minneapolis (2001)
11. Karypis, G.: CLUTO. A clustering toolkit. Technical Report 02–017, University of Minnesota, Department of Computer Science (2003)
12. Rohde, D.: SVDLIBC, http://tedlab.mit.edu/~dr/svdlibc

Hybrid Immune Algorithm for Many Optima

Małgorzata Lucińska

Kielce University of Technology
Al. 1000-lecia PP. 7, 25314 Kielce, Poland

Abstract. In this paper an algorithm is proposed, which combines elements of immune network and clonal selection together with gradual narrowing of a search area. It is used for optimization of multi-modal functions and enables finding many optima in given domains. The algorithm introduces a novel way of interaction between memory cells and population cells. The influence of cell interaction strength on the algorithm performance has been investigated. Experiments prove that the algorithm is capable of fast localization of many optima. It outperforms other presented approaches to the multi-modal function optimization problem.

Keywords: Artificial immune systems, clonal selection, function optimization.

1 Introduction

This paper investigates a modification of a Hybrid Immune Algorithm presented in [1]. HIA is meant to find one optimum for multi-modal function with a use of immune metaphor combined with gradual narrowing of a search area. HIAMO (Hybrid Immune Algorithm for Many Optima) constitutes an extension of HIA, which allows for detection of many optima for multi-modal function.

There are many solutions to optimization problems, which concentrate on finding one global optimum of multi-modal functions. Finding many optima seems to be less popular task, which requires some modifications to the former solutions, in order to prevent the system individuals from gathering in just one optimum. Niching [2] is a policy, used in biologically inspired algorithms, which allows a population to divide into smaller groups and search different areas in parallel. The technique has been successfully used in order to detect many optima of multi-modal functions, for example [6].

In this paper another solution is proposed, which is based on immune metaphor. Immune inspired algorithms [3] use some elements, functions and mechanisms characteristic for natural immune systems. They seem to be a very suitable approach for multi-modal function optimization, because of their main features, like distribution and adaptability. The task of the natural immune system is to identify and destroy foreign invaders or antigens. The natural immune system is composed of lymphocytes — B lymphocytes (B-cells) and T lymphocytes (T-cells). B-cells produce antibodies, which bind to the invading antigens and help destroy them.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 540–547, 2010.
© Springer-Verlag Berlin Heidelberg 2010

Each B-cell produces only one kind of antigenic receptor. When an antigen enters the body, it activates only the lymphocytes whose receptors can bind to it. Activated by an antigen and with a second signal from accessory cells, such as the T-cells, the B-cells proliferate (divide) producing a large number of clones. In the final stage these clones can mutate in order to produce antibodies with very high affinity to a specific antigen. The process is explained by the clonal selection principle, according to which only those cells that recognize the antigens are selected to proliferate. The production of new cells, mutation and selection is known as maturation of immune response, because it allows the immune cells to respond better to known antigens.

One of the widely established immune optimization algorithms is opt-aiNet, consult [4] for details. The main goal of the algorithm is to find all local and global function optima and to maintain them. It creates a population of antibodies, which are evaluated against the objective function (affinity determination) and cloned proportionally to their affinity. Then cells undergo a process of mutation. Clones with the highest affinity are placed into a clonal memory set. The chosen clones interact with each other, which results in suppression of the individuals, whose affinity with others is less than a specified threshold. HIAMO uses the same interactions as opt-aiNet, but also adds new effects. It introduces repulsion between population cells and memory cells. In this paper the influence of the interaction on the algorithm effectiveness is investigated.

In Section 2 the components of the system are introduced and HIAMO steps are explained. Then, in Section 3, I show how new interaction parameters affect the algorithm performance and compare it to other solutions. Finally, in Section 4, the main conclusions are drawn.

2 Hybrid Immune Algorithm for Many Optima

Similary to HIA, HIAMO employs a very similar system model and uses the same terminology as the opt-aiNet algorithm [4]:

- Cell (antibody): individual, which is described as a real-valued vector in a Euclidean shape-space;
- Fitness: the fitness of a cell equals the value of the objective function in a given point with coordinates representing the given cell;
- Affinity: Euclidean distance between two cells;
- Clone: offspring cells, that are identical copies of their parent cell. They undergo somatic mutation in a further step.

The algorithm creates two types of memory sets: one temporary memory set for each individual in the population, which includes its fittest descendant found during a current epoch, and one permanent memory set S for the whole population, containing the best candidate solutions found so far. A number of temporary memory sets equals a number of cells in the population. A value of population size is discussed in the next section of the paper. For the sake of clarity let's assume that the algorithm searches for objective function maxima.

HIAMO pseudocode

```
Until algorithm termination condition is not met do:
 I Randomly generate population P
 II Until epoch termination is not met:
   1. For each p belonging to P:
     a. If age of cell p is smaller than w
       i. Produce |c| clones and mutate them
       ii. Find the fittest clone cp
       iii. If f(cp)>f(p) replace p by cp
       iv. Else increase age of element p
     b. Else
       i. Move cell p to its temporary memory set Mp
       ii. Perform suppression of the set Mp
       iii. Limit size of temporary memory to |m| cells
       iv. Generate new population cell p
   2. Move the best cell in all sets Mp to set S
   3. Remove all elements from temporary memory sets.
   4. Perform repulsion between set S and population cells
```

Cells are subject to ageing. Each time an antibody has produced clones less fit than itself, its age increases by one. When the age reaches an established value w, the cell is moved to the temporary memory set. Similar policy is used in the opt-IMMALG algorithm [5]. The authors have introduced so called aging operator, which eliminates old cells in a population. The mechanism allows for maintaining high diversity in order to avoid premature convergence. The difference between the two algorithms lies in a possibility of further use of the old antibodies as memory cells in HIAMO.

Mutation of the clones is performed with the help of two operators. The first one uses Gaussian distribution independently for each dimension. The following formula describes the process:

$$c_k = N(p_k, \alpha_k) \tag{1}$$

$$\alpha_k = 2(p_{k-1} - c^p_{k-1}) \qquad \text{if } (c^p_{k-1}) < f(p_{k-1}) \tag{2}$$

where c_k and p_k are vectors of coordinates describing appropriately a mutated cell and a parent cell in k-th iteration, $N(p_k, \alpha_k)$ is a Gaussian random variable of mean p_k and standard deviation α_k. If the best clone fitness is better than that of the parent, the standard deviation is a vector with coordinates proportional to the difference of coordinates between the parent cell and its best clone in the previous iteration. Otherwise the standard deviation remains the same as in the previous iteration.

The second way of mutation concerns only one dimension, the other coordinates of the mutated clone remain the same as of the parent cell. The dimension is chosen randomly and a new value of the coordinate is generated with the help of uniform distribution within the appropriated ranges for this dimension. The second sort of mutation is carried out in 20% of all cases, which are also chosen at random. In the first phase of the algorithm operation both types of mutation allow for exploration of a search space. Whereas at latter stages the first type of mutation is mainly focused on exploitation but the second one still enables exploration, which avoids getting stuck in a local extreme.

In the next step of the algorithm the mutants are evaluated against the objective function to be optimized. The best individual is selected from all the mutants produced by the cell. Fitter descendants replace their parents. For each dimension a change of the coordinate value is calculated, which equals a difference between two coordinate values — one of the newly chosen best cell in the generation and the second one of the old parent. The change value will be used in the next iteration as a standard deviation in the Gaussian distribution for mutation purpose ([2]).

After reaching a given age an antibody is moved to a temporary memory set. Cells included in the same temporary memory set interact with each other similarly as in opt-aiNet. If affinity between two individuals is smaller than the suppression threshold σ the worse one is removed from the memory. In addition each temporary memory set has a size $\mid m \mid$ limited to a few of the fittest cells.

The antibody, which was moved to the temporary memory set, is replaced by a new individual. At the first stage of the algorithm the new cells are generated randomly within the whole search space. When the size of the temporary memory reaches its specified value $\mid m \mid$ newcomers are created with a use of Gaussian distribution:

$$a_k^i = N(b_k^i, \sigma_k^i) \qquad \text{for } i \in (1, D) \tag{3}$$

$$\sigma_k^i = \beta_k \sigma_{k-1}^i + (1 - \beta_k) \Delta_k^i \tag{4}$$

$$\Delta_k^i = max\{(m_{kc}^i - m_{kd}^i); \qquad c, d \in (1, \mid m \mid)\} \tag{5}$$

$$\beta_k = \beta_0 (1 - (1 - 1/k)^q) \tag{6}$$

where a_k^i is i-th coordinate of the new cell, generated in k-th iteration; σ_k^i denotes standard deviation of the distribution; b_k^i stands for the mean value and equals i-th coordinate of the best cell in the temporary memory; β_k is a learning factor in k-th iteration; $\beta_0 = 0.8$ and $q = 5$; Δ_k^i describes the maximum difference between any two memory cells coordinates; m_{kc}, m_{kd} represent cells of the temporary memory.

In the later phase of the execution of HIAMO, the search area of each population cell becomes more and more narrowed. Exploration gradually gives way to exploitation. A standard deviation of the Gaussian distribution, responsible for reduction of the search area, is controlled by a learning factor β_k. It prevents too fast convergence to a local extreme. If during two subsequent iterations, after changing values of distribution parameters, generated cells have smaller fitness than the best temporary memory cell, the new values are replaced by the old ones. Such a policy avoids exploitation of misleading areas.

A novelty of HIAMO is the repulsion between population cells and cells creating the permanent memory set. If affinity between such two cells is smaller than a specified threshold θ, coordinates of the first one are changed at random, in order to increase their mutual distance. This type of interaction avoids focusing on one function optimum and increases exploration of the whole search area, which helps to find many function optima faster.

The epoch termination condition can be based upon an achievement of the optimum value or a lack of improvement of a fitness value for given number of iterations. The algorithm runs until the specified number of optima is found or the number of optima does not increase during a determined number of iterations.

3 Experiments

The above algorithm has been implemented in Java language. The performance of HIAMO was investigated with a use of well-known benchmark functions published in [6] and [7]. All the benchmark functions are multi-modal and non-separable. They cover a wide range of landscape types to allow for assessment of the algorithms' performance in different conditions. The functions and their domain ranges are shown in Table 1. In the experiments the goal of the algorithm was to locate a specified number of global optima with an accuracy of ϵ=0.00001 for functions *f1-f5* and ϵ=0.1 for function *f6*. The presented results are an average value calculated on the basis of 1000 independent experiments, unless indicated differently. The following parameter values have been employed: number of clones $\mid c \mid$=7, suppression threshold σ=0.00002, temporary memory size $\mid m \mid$=4, and maximum cell age w=4. Parameter tuning relies on experimental comparisons.

Table 1. Functions for experiments

Id	Function	Parameters
f1	$(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})cos(x_1) + 10$	$x_1 \in [-5, 10], x_2 \in [-10, 10]$
f2	$(-4[(4 - 2.1x_1^2 + \frac{x_1^4}{3})x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2]$	$x_1 \in [-2, 2], x_2 \in [-1.1, 1.1]$
f3	$sin^6(5\pi x)$	$x \in [0, 1]$
f4	$200 - (x_1^2 + x_2 - 11)^2 - (x_1 + x_2^2 - 7)^2$	$x_1, x_2 \in [-6, 6]$
f5	$\sum_{j=1}^{5} jcos((j+1)x_1 + j) \sum_{j=1}^{5} jcos((j+1)x_2 + j)$	$x_1, x_2 \in [-10, 10]$
f6	$\sum_{j=1}^{3} exp\frac{-\sum_{i=1}^{n}(x_i - a_j)^2}{0.09}$	$x_i \in [-2, 2], \boldsymbol{a} = (-1, 0, 1)$

As a novelty of HIAMO lies in the introduction of repulsion between population cells and permanent memory cells, first an influence of the repulsion threshold θ on the algorithm effectiveness is investigated. For each function the minimum θ value was chosen so that the change in the function value within the distance of θ is bigger than the accuracy of optima determination. Such condition avoids treating one peak as two separate optima. The maximum value of repulsion threshold depends on the minimum distance between the optima of a given function. Pictures 1-2 show for two benchmark functions, how number of function evaluations, necessary to locate all optima, depends on θ value. The experiments were done for different population sizes, varying from one to ten. Worth noticing is the fact, that regardless of a used function, the best results are achieved for a population consisting of only one cell. Traditionally in immune algorithms population includes many antibodies. In case of multi-modal function, cells searching for an optimum often finish on the same peak, generating useless redundancy and increasing number of function evaluations. For population sizes bigger than a number of function optima, efficiency of the algorithm decreases with an increase of the repulsion threshold. This means that the effect of multiple detection of one optimum by many cells is bigger than repulsion influence and leads to an enormous number of function evaluations. Other regularities appear for smaller population sizes becoming quite distinct for one cell population. At the beginning the number of function evaluations decreases with an increase of repulsion threshold values and then it stabilizes in order to dramatically increase at the end. When θ is small the interactions occur very rarely, only when cells are very close to each other and exploitation of the already found optima is not avoided. On the other hand, interactions between remote elements, resulting in over intensive repulsion of cells, prevents detection of adjacent optima. The best policy for each benchmark function would be to set the θ value at about 0.1. The results speak in favor of cell repulsion introduction and limit of population size to one antibody.

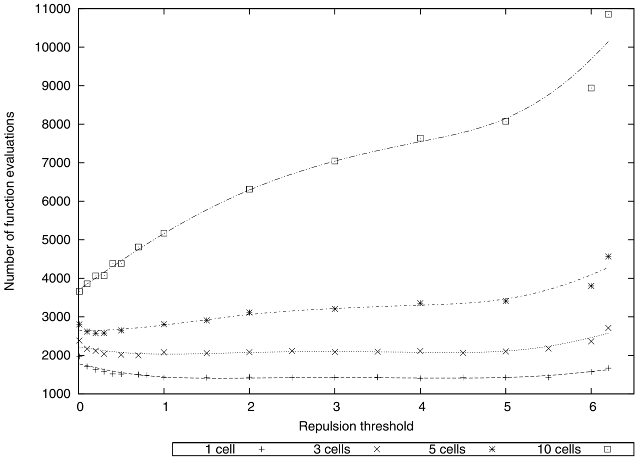


Fig. 1. Performance of HIAMO for function *f1* using different values for repulsion threshold and different population sizes. Function *f1* has 3 global optima.

The performance of HIAMO was compared with other algorithms. As benchmark solutions I have used the following algorithms: opt-aiNet, kPSO, and SDE with switching strategies. The first one is an established immune algorithm, mentioned in the first section of this paper. The second one belongs to a class of particle swarm optimization algorithms [6]. The last algorithm represents

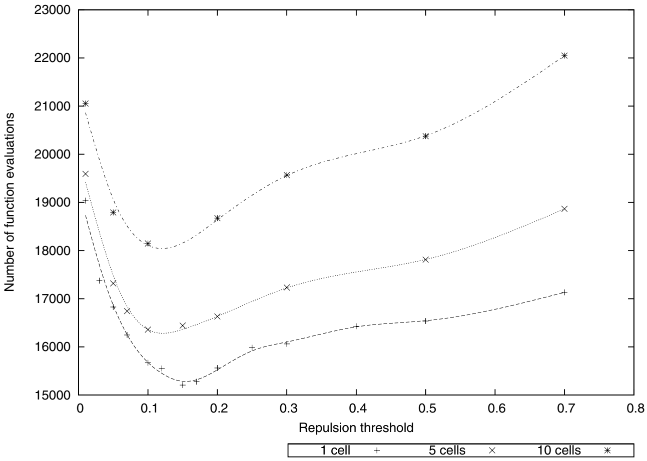


Fig. 2. Performance of HIAMO for function *f5* using different values for repulsion threshold and different population sizes. Function *f5* has 18 global optima.

Table 2. Performance of opt-aiNet, kPSO, SDE with switching strategies (SDEwss), and HIAMO (*d* indicates problem dimension)

Id	*d*	Number of function evaluations			
		opt-aiNet	kPSO	SDEwss	HIAMO
f1	2	25560 ± 14718	2080 ± 440	4350 ± 380	1700 ± 814
f2	2	3420 ± 1424	1120 ± 216	2000 ± 276	440 ± 205
f3	2	1000 ± 552	1200 ± 688	—	1000 ± 410
f4	2	15670 ± 9658	2260 ± 539	6490 ± 544	1780 ± 914
f5	2	82700 ± 50900	81190 ± 45460	16300 ± 2320	18490 ± 5974
f6	3	2306 ± 625	—	2250 ± 245	585 ± 226
f6	5	5600 ± 1000	—	2880 ± 207	1220 ± 420
f6	10	10670 ± 2260	—	10900 ± 13000	4450 ± 2280

species-based differential evolution techniques [7]. In the experiments, repulsion threshold θ for functions *f1*-*f5* equals 0.1, whereas for *f6* — 0.5. Parameters for opt-aiNet, kPSO and SDE with switching strategies are given respectively in [4], [6], and [7]. As can be seen from the results included in Table 2, HIAMO outperforms the other algorithms in terms of number of evaluations, being equal to them in terms of the quality of solutions. The differences in some cases are dramatically high, for example for *f5* function the HIAMO result is about 5 times better than that for kPSO and opt-aiNet. The number of evaluations obtained for SDE with switching strategies is the smallest one in the case of function *f5*, but the accuracy of the solution is one thousand times worse than for the other algorithms.

4 Conclusions

This paper has presented a novel hybrid algorithm HIAMO, introducing two sorts of memory sets and a new way of interaction between population cells and memory cells. Thanks to the interactions the algorithm does not focus on one extreme but is capable of exploring a whole domain in search for other solutions. Experiments carried out on the benchmark functions show that HIAMO is competitive with other algorithms using biologically inspired techniques.

References

1. Lucińska, M., Wierzchoń, S.T.: Hybrid Immune Algorithm for Multimodal Function Optimization. In: Recent Advances in Intelligent Information Systems, pp. 301–313. EXIT, Warszawa (2009)
2. Deb, K., Goldberg, D.E.: An investigation of niche and species formation in genetic function optimization. In: International Conference Genetic Algorithms, pp. 42–50. Morgan Kaufmann, San Francisco (1989)
3. De Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Approach. Springer, Heidelberg (2002)
4. De Castro, L.N., Timmis, J.: An Artificial Immune Network for Multimodal Function Optimisation. In: IEEE World Congress on Evolutionary Computation, pp. 669–674. IEEE Press, New York (2002)
5. Cutello, V., Narzisi, G., Nicosia, G., Pavone, M.: An Immunological Algorithm for Global Numerical Optimization. In: Talbi, E.-G., Liardet, P., Collet, P., Lutton, E., Schoenauer, M. (eds.) EA 2005. LNCS, vol. 3871, pp. 284–295. Springer, Heidelberg (2006)
6. Passaro, A., Starita, A.: Particle swarm optimization for multimodal functions: a clustering approach. Journal of Artificial Evolution and Applications 8(2), 1–15 (2008)
7. Shibasaka, M., Hara, A., Ichimura, T., Takahama, T.: Species-Based Differential Evolution with Switching Search Strategies for Multimodal Function Optimization. In: IEEE Congress on Evolutionary Computation, pp. 1183–1190. IEEE Press, Los Alamitos (2007)

Combining ESOMs Trained on a Hierarchy of Feature Subsets for Single-Trial Decoding of LFP Responses in Monkey Area V4

Nikolay V. Manyakov[1], Jonas Poelmans[2],
Rufin Vogels[1], and Marc M. Van Hulle[1]

[1] Laboratory for Neuro- and Psychofysiology, K.U.Leuven,
Herestraat 49, bus 1021, 3000 Leuven, Belgium
[2] Faculty of Business and Economics, K.U.Leuven,
Naamsestraat 69, 3000 Leuven, Belgium
{NikolayV.Manyakov,Rufin.Vogels,Marc.VanHulle}@med.kuleuven.be,
Jonas.Poelmans@econ.kuleuven.be

Abstract. We develop and combine topographic maps trained on different combinations of feature subsets for visualizing and classifying event-related responses recorded with a multi-electrode array chronically implanted in the visual cortical area V4 of a rhesus monkey. The monkey was trained, during consecutive training sessions, in a classical conditioning paradigm in which one stimulus was consistently paired with a fluid reward and another stimulus not. We opted for features from three categories: time-frequency analysis, phase synchronization between electrodes, and propagating waves in the array. The Emergent Self Organizing Map (ESOM) was used to explore the feasibility of single-trial decoding. Since the effective dimensionality of the feature space is rather high, a series of ESOMs was trained on features selected from different combinations of the three feature categories. For each trained ESOM, a classifier was developed, and classifiers of different ESOMs were combined so as to maximize the single-trial decoding performance.

1 Introduction

An event-related potential (ERP) is any stereotyped electrophysiological response to a stimulus. The last few years have witnessed an increasing interest in detecting ERPs, due to the development of brain-computer interfaces (BCIs). The problem of ERP detection requires a compromise between classification accuracy and response time, pushing the signal-to-noise ratio requirements to its limits. Indeed, in the ideal case, we would like to be able to detect an event based on a single ERP brain response, thus, without averaging. However, in practice, this is not very likely, since the ERP is a small signal in the ongoing brain activity. As a result, multiple presentations of the targeted stimulus are needed, and the corresponding time-locked responses averaged.

In this paper, we will deal with event-related local field potentials (LFPs) that represent the extracellular current flow due to the summed postsynaptic

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 548–555, 2010.
© Springer-Verlag Berlin Heidelberg 2010

potentials of local cell groups [1]. The LFPs were recorded with a 96 electrode array implanted in the visual cortical area V4 of a rhesus monkey. The monkey was shown two stimuli, one for which he received a fluid reward, and another not (classical conditioning paradigm). It can be seen that the variability in the single LFP responses is large, and the difference between the responses to the two stimuli is not visible. But if we compute the averaged ERPs, we observe a clear difference in amplitude. As was shown in [2], the difference in the averaged ERP for the rewarded and unrewarded stimuli grows as a result of training, due to brain plasticity, from being identical in the beginning to markedly different at the end. But the question whether it is possible to detect differences in single trial recordings (thus, without any time-locked averaging) remains open. This question is important because it opens the way to perform real-time BCI. In this paper, we will concentrate on the possibility to detect and visualize differences in single trial LFP recordings using the Emergent Self Organizing Map (ESOM) [4].

The ESOM is a topographic map that differs from the traditional Self Organizing Map (SOM) [3] in that a very large number of neurons (at least a few thousands) is used [4]. According to [5], "emergence is the ability of a system to produce a phenomenon on a new, higher level." In order to achieve emergence, the existence and cooperation of a large number of elementary processes is necessary, so that large numbers of neurons can represent data clusters individually, which facilitates their detection. The ESOM is argued to be especially useful for visualizing sparse, high-dimensional datasets, yielding an intuitive overview of its structure [6]. In this paper we use, for the first time, the ESOM for visualizing and decoding intracranial recordings.

2 Experiment

We briefly summarize the experimental set-up; for more information, we refer to [2]. A rhesus monkey was implanted with a Utah array into the prelunate gyrus (area V4). The array measures 4 × 4 mm, and consists of 10 × 10 electrodes (4 of them are not connected). The local field potential signals were obtained by filtering the recorded signals between 0.3-250 Hz. The stimuli are displayed in Fig. 1. During training, a different sinusoidal noise background, that filled the display, was presented every 500 ms. At random intervals, a sinusoidal grating

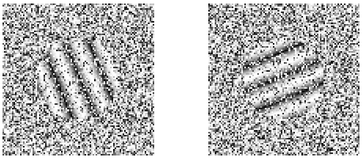


Fig. 1. Examples of rewarded (left panel) and unrewarded (right panel) stimuli. The sinusoidal grating has an orientation 157.5^o (rewarded) or 67.5^o (unrewarded), a spatial frequency of 2 c/deg, a diameter 4^o visual angle, and is superimposed on a sinusoidal noise background (S/N ratio=20%).

was presented for 500 ms. The reward was provided 400 ms after presentation of the grating pattern and, thus, partially overlapped with the grating presentation.

For our analysis, we looked only into the period of 300 ms after stimuli onset (thus, 100 ms before reward), to avoid the influence of the reward, and concentrate on ERP signals. Also, only those recordings of the rewarded and unrewarded stimuli were considered that were preceded by 3 background images, and that were observed by the monkey without failing to fixate a small dot on the screen. In this way, on average, 262 rewarded and 282 unrewarded trials per day were retained for further analysis.

3 Emergent Self Organizing Map

An Emergent Self Organizing Map (ESOM) map is composed of a set of formal (artificial) neurons I, arranged in a lattice or map. A neuron $i \in I$ is a tuple $(\mathbf{w}_i, \mathbf{x}_i)$ consisting of a weight vector $\mathbf{w}_i \in \mathbb{R}^n$ and a position $\mathbf{x}_i \in \mathbb{R}^2$ in the lattice. The input space F is a metric subspace of $\mathbb{R}^n$. The training set $F^l = \{\mathbf{f}_1, ..., \mathbf{f}_N\}$ with $\mathbf{f}_1, ..., \mathbf{f}_N \in \mathbb{R}^n$ consists of input samples presented during ESOM training. We perform online training in which the best match for an input vector is searched for, and the corresponding weight vectors, and also those of its neighboring neurons of the map, are updated immediately.

When an input vector $\mathbf{f}_k$ is supplied to the training algorithm, the weight of neuron i is modified as follows: $\Delta\mathbf{w}_i = \eta \cdot h(i, bm_k, R) \cdot (\mathbf{f}_k - \mathbf{w}_i)$, with $\eta \in [0, 1]$, and bm_k the best-matching neuron of an input vector $\mathbf{f}_k$, *i.e.*, $bm_k = arg\min_{i \in I} \|\mathbf{f}_k - \mathbf{w}_i\|$, where $\|\mathbf{f}_k - \mathbf{w}_i\|$ is a distance in input space, according to some metric, between the input vector $\mathbf{f}_k$ and the weight vector $\mathbf{w}_i$ of the ith neuron; $h(i, bm_k, R)$ is a function that scales the update of neuron i according to its position in a neighborhood with radius R (according to some metric in the map) of the best-matching neuron bm_k. The ESOM training and visualization is done with the publicly available Databionic ESOM tool [7].

4 Features

The objective of feature selection is three-fold: 1) to improve the prediction performance of the classifier, 2) to provide classifiers with low computational complexitythat are also more cost-effective, and 3) to yield a better understanding of the underlying process that generated the data [8]. We considered the next feature subsets:

4.1 Time-Frequency Analysis

For the time time-frequency analysis, we used the continuous wavelet transformation (CWT) of the LFP signal $s(t)$, defined as:

$$W(a, b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} s(t) \cdot \overline{\psi\left(\frac{t-b}{a}\right)}\, dt \qquad (1)$$

where b is a time shift, a is a scale factor and ψ is the predefined mother wavelet (we use the Daubechies wavelet db7) with zero mean, $\int_{-\infty}^{\infty} \psi(t)\, dt = 0$.

4.2 Phase Synchrony

As a second feature, we used an index of phase synchrony between the recorded LFPs, which was shown to be useful for decoding LFP recordings [9]. The phase from the prefiltered LFP signal $s(t)$ (in our case filtered below 30 Hz) is estimated according to $\varphi_s(t) = arctan\left(\frac{H\left(s(t)\right)}{s(t)}\right)$, where $H(s(t))$ is the Hilbert transform of signal $s(t)$. As an index of bivariate phase synchrony between signals $x(t)$ and $y(t)$, we use the mean phase coherence, estimated as $\gamma_{x,y} = \sqrt{|\langle e^{i(\varphi_x(t)-\varphi_y(t))}\rangle|}$, where $\langle\cdot\rangle$ denotes the time average, and which measures how the relative phase difference is distributed over the unit circle. The result is in the interval $[0, 1]$, where 1 correspond to perfect synchrony.

4.3 Propagating Waves

As a third set of features, we took the directions and velocities of waves propagating in the array, since the presence of such phenomena was shown in [9,10]. In order to detect propagating waves, we used a phase-based estimate of the optical flow [11], in which case the electrode array should be regarded as a 10×10 pixel image, with each electrode corresponding to a pixel. By coupling the grey level of each pixel to the LFP of the corresponding electrode, the array becomes a frame in an image sequence (movie). To estimate the optical flow, the array's LFP recordings become a movie in the form $s(x, y, t)$, where x and y refer to the pixel's position and t refers to time. The previously derived phases (see 4.2) are now referred to as $\varphi(x, y, t)$. The velocity of the coherent activity in the array is defined as the velocity of the lines of constant phase [11]. This velocity $\mathbf{v} = (dx/dt, dy/dt)$ is computed by taking the total derivative of $\varphi(x, y, t) = C$ with respect to time $d\varphi/dt = \bigtriangledown\varphi \cdot \mathbf{v} + \partial\varphi/\partial t$. The velocity direction, which is perpendicular to the lines of constant phase, is $- \bigtriangledown \varphi$.

5 Results

We considered, for our analysis, all electrodes for the phase synchrony and the wave features, but for the wavelet features only those of two electrodes (#18 and #80) that are representative for two groups of electrodes with similar LFP responses. As time-frequency features, we divide the wavelet scalograms (for 300 ms analyzed recordings, we applied a transformation for the scale factor a ranging from 1 to 300) into 20×20 squares to avoid that too many redundant features would be taken. Within each square, the best separable features, based on the t-statistic, were chosen. Then, from the resulting 225 features, only the 15 best features (with smallest p-values) are retained. Hence, for the two electrodes considered, we have 15 best separable features for each electrode.

For the phase synchrony, we calculate the level of synchrony between all electrode pairs within non-overlapping windows of 20 ms (so, for each electrode pair, we have $300/20 = 15$ synchrony features). These features are then sorted in terms of the separability between the two stimuli using the t-statistic. Only the 15 best phase synchrony features are retained.

As wave propagation features, the direction and magnitude of the optical flow in each electrode, for each moment after stimulus onset, are calculated. Among these features, also the 15 best separating ones are retained.

In total, we have 60 features for decoding. These features are developed for every training day and used for training separate ESOMs, not only for visualization purposes, but also for decoding the responses to the two stimuli. To test the decoding performance, the ESOM trained on a given day was used for predicting the decoding performance for the next day.

For days 1–36 of the monkey's training, we used the 2D ESOM with a toroid architecture sized 50×82 formal neurons, given all 60 features. ESOMs were trained for 20 presentations of the training set. As a distance metric, we used the Euclidian distance, both in data space and in lattice space. For every training sample, the weights of the winning neuron, and of the neurons in its neighborhood, were updated. The neighborhood function was a gaussian kernel. The neighborhood radius was linearly decreased during training from 24 to 1, and the learning rate from 0.5 to 0.1. Weights in the network were initialized by sampling the normal distribution. As a preprocessing, we transferred all features into Z-scores.

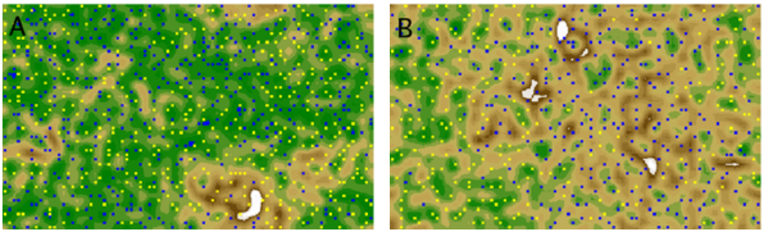


Fig. 2. ESOMs for monkey training day 1 (A) and day 36 (B) using all 60 features. The responses to the rewarded stimulus are shown as blue pixels, and those to the unrewarded one as yellow pixels. The background shows, for each neuron, the average distance of its weight vector to those of its lattice neighbors, normalized over all average distances in the lattice (U-matrix) (green: smallest distance; white largest distance).

In Fig. 2 A,B, the ESOMs for the first and last days of the monkey's training are shown. It can be seen how clusters for two separate groups were formed as a result of monkey training (note that no labeled information was used in training the ESOMs). As a measure of how well the clusters are formed, we opted for the standard deviation (in terms of distances) of the points in each class in a map X. Note that Euclidian distances were calculated by taking into account the toroidal structure. We plotted the mentioned measure as a function of the training days, for each stimulus [results not shown]. The result is that the standard deviation decreases over time, for both the rewarded and the unrewarded stimuli, which indicates that the clusters become more condensed.

We also calculated the average Euclidean distance between two consecutive responses to the same stimulus in the original feature space F [results not shown]. We obtained that the average distance decreases over time for the rewarded stimulus, but not for the unrewarded one. This means that the responses to the consecutive rewarded stimulus vary less than to the unrewarded stimulus.

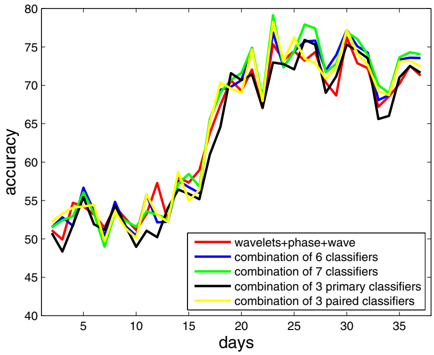


Fig. 3. Prediction accuracy of responses based on the map, trained in the previous day, for different combinations of classifiers

We also computed the prediction performance of the developed ESOMs, given all 60 features. For this, we used the weighted kNN classifier:

$$a(u; X^l, k) = arg \max_{y \in Y} \sum_{i=1}^{k} [y_{i,u} = y]\omega_i, \tag{2}$$

where u is the coordinate of the best-matching neuron in the ESOM for each new item we want to classify; X^l is the set of coordinates of the best-matching neurons for training examples F^l; $Y = \{-1, 1\}$ is the set of class labels (rewarded and unrewarded classes); $\omega_i = q^i$ $(q = 0.9)$ is a weight given to the ith closest labeled sample in the training set X^l; $y_{i,u}$ is the label of the ith closest sample with respect to u; $[a = b]$ is 1 when $a = b$ and 0 elsewhere. The parameter k was determined using leave-one-out cross-validation on the training set.

Using the ESOM constructed for the previous day, we predict the stimulus type (rewarded or unrewarded) that corresponds to the response recorded on the current day. The results are shown in Fig. 3 (red line). We observe the increase in performance as a function of training days, which indicates the ESOM's potential to represent and distinguish different types of stimuli based on single trial recordings.

Hierarchy of Feature Subsets

One can question whether the projection onto a 2 dimensional ESOM is valid. To this end, we calculated the inner or effective dimension of the manifold consisting of the data point in feature space F. We opted for the Grassberger-Procaccia correlation dimension [12]. For points $\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_N$ in our preselected feature space with the Euclidian distance between the points $\|\mathbf{f}_i - \mathbf{f}_j\|$, and for a positive number r (radius), we estimated the correlation sum $C(r)$ as the fraction of pairs whose distance is smaller than r according to:

$$C(r) = \frac{2}{N(N-1)} \sum_{i<j} \Theta(r - \|\mathbf{f}_i - \mathbf{f}_j\|), \tag{3}$$

where $\Theta(x)$ is the Heaviside step function. From the known relation $C(r) \sim r^D$, one can estimate the manifold dimension D as the slope of the straight part of the *log-log* plot. After determining the dependency of $C(r)$ on r, and calculating the manifold dimension for all training days, we found that the dimensions of the rewarded and unrewarded stimuli are always similar and decrease over time (from 19.078 in the first day to 12.202 in the last training day).

Since the effective dimension is always above 2, we can question the faithfulness of the projection onto a two-dimensional ESOM. One way to address this problem, without loosing the advantage of a two-dimensional visualization, is to develop several ESOMs, trained on particular subsets of features, and to combine their classification outcomes. Hence, we used a hierarchical feature grouping strategy, and trained ESOMs on wavelet-, phase-, and wave features separately (primary case, thus, 3 ESOMs), but also ESOMs trained on pairs of features (wavelets+phases, wavelets+waves, waves+phases, thus, 3 ESOMs). We consider the following possible combinations of classifiers: 1) the primary classifiers (combination of 3 primary classifiers), 2) the classifiers for the pairs of features (combination of 3 paired classifiers), 3) the classifiers for the primary and the paired features (combination of 6 classifiers), and 4) the classifiers for the primary-, paired-, and all features (combination of 7 classifiers).

For every trained map, for a particular day, the optimal value of k in the kNN classifier was estimated by means of a leave-one-out cross-validation. The optimal value of k for the jth ESOM, is defined as k_o^j, and the corresponding maximum leave-one-out cross-validation performance is defined as M_j. The outcomes of the classifiers of the different ESOMs were combined as follows. For any new sample to be classified, we estimate $a_j(y) = \sum_{i=1}^{k_o^j} [y_{i,u} = y]\omega_i$ (notation: see above) for each ESOM considered in the combination. Based on these results, we assign the outcome of $V_j(y) = \dfrac{|a_j(1) - a_j(-1)|}{a_j(1) + a_j(-1)}$ to the class label y for which $a_j(y)$ is maximal, and $V_j(y) = 0$ to the second class, for all ESOMs that are combined. The resulting class label for the new sample, based on the combined classifiers, is then defined as follows: $arg\max_{y \in Y} \sum_{j=1}^{n} M_j V_j(y)$. The prediction performance for the current day samples, based on the combined classifiers and ESOMs constructed for the previous day, for different combinations of feature sets, are shown in Fig. 3. It can be observed that the performance for some combinations of feature sets is significantly (based on ANOVA) better than that for the whole set of features (from day 18 onwards): the combination of 7 classifiers is significantly different from the classifier based on the full set of features ($p = 7.6 \cdot 10^{-5}$); the combination of 6 classifiers also ($p = 1.2 \cdot 10^{-3}$), and combination of the paired classifiers also ($p = 2 \cdot 10^{-3}$).

6 Conclusion

We have used the ESOM for visualizing and classifying event-related responses recorded with a multi-electrode array implanted in area V4 of a rhesus monkey. Since the effective dimensionality of the feature space is still quite high, one can question the relevance of using a two-dimensional ESOM. The solution we propose is to combine the classification outcomes of ESOMs trained on various feature subsets. We have shown that, in this way, a significantly better decoding performance is obtained.

Acknowledgments. NVM is supported by IST-2004-027017. MMVH is supported by research grants EF 2005, CREA/07/027, G.0588.09, IAP P6/29, GOA 2000/11, and IST-2007-217077.

References

1. Buzsáki, G.: Large-scale recording of neuronal ensembles. Nature Neuroscience 5, 446–451 (2004)
2. Frankó, E., Seitz, A.R., Vogels, R.: Dissociable Neural Effects of Long-term Stimulus-Reward Pairing in Macaque Visual Cortex. Journal of Cognitive Neuroscience (to appear)
3. Kohonen, T.: Self-organizing maps. Springer, Heidelberg (1995)
4. Ultsch, A., Hermann, L.: Architecture of emergent self-organizing maps to reduce projection errors. In: Proc. ESANN 2005, pp. 1–6 (2005)
5. Ultsch, A.: Data Mining and Knowledge Discovery with Emergent Self-Organizing Feature Maps for Multivariate Time Series. In: Kohonen Maps, pp. 33–46 (1999)
6. Ultsch, A.: Density Estimation and Visualization for Data containing Clusters of unknown Structure. In: Proc. GfKl 2004 Dortmund, pp. 232–239 (2004)
7. Databionic ESOM Tools, http://databionic-esom.sourceforge.net/
8. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research 3, 1157–1182 (2003)
9. Manyakov, N.V., Van Hulle, M.M.: Synchronization in monkey visual cortex analysed with information-theoretic measure. Chaos 18, 037130 (2008)
10. Rubino, D., Robbins, K.A., Hatsopoulos, N.G.: Propagating waves mediate information transfer in the motor cortex. Nature Neuroscience 9(12), 1549–1557 (2006)
11. Fleet, D.J., Jepson, A.D.: Computation of component image velocity from local phase information. Int. J. Comput. Vis. 7, 77–104 (1990)
12. Grassberger, P., Procaccia, I.: Measuring the strangeness of strange attractors. Physica D 9(1-2), 189–208 (1983)

XML Schema and Data Summarization

Jakub Marciniak

Faculty of Mathematics and Computer Science,
Adam Mickiewicz University, Poznan, Poland
kubam@amu.edu.pl

Abstract. As XML repositories are becoming more and more complex there is a need to develop methods and tools to facilitate the understanding and exploring schemas and contents of these repositories. A solution can be provided by a proper summarization of XML documents. In this paper we propose the summarization concerning both the schema and the contents of XML documents. There are three general steps in our approach: (1) the schema is extracted from a given XML document; (2) a summary of the schema is derived, and correspondences between the summary and the underlying source schema are established; (3) the summarization information is used to summarize (aggregate) contents (text values) of instance document. We show how the user can be involved in this process. We develop new algorithms used in the summarization process. We show that our approach is useful and effective in practice.

Keywords: XML summarization, XML transformation, Schema extraction.

1 Introduction

A summary for a given complex XML schema is such a schema that provides the users with a concise overview of the original schema for better understanding. Each element in the summary represents a cluster of the original schema. Thus, the challenging problem is the selection of such elements that are the most important for the understanding the original schema. In this paper we discuss a method of selecting most important nodes in a schema based on the adaptation of well-known PageRank algorithm. In this approach, a weight is assign to each edge in the schema. Weights of all edges adjacent to a node are then used to calculate the weight of the node (label of the schema). Next in the paper we observe that even a well-summarized schema may be insufficient for proper understanding of the semantics of the underlying schema. To facilitate this, we propose a method for *data summarization*. It is easier for the user to understand a schema if he is provided not only with good schema summary but also with sample data conforming to this summary. We show a way for obtaining data summary in accordance with schema summary. We propose so-called *summary functions* which combine *horizontal* and *vertical* components. A horizontal component is a function iterating over elements of a set and returning an aggregation of these elements. The vertical components iterates along hierarchy of sets. We developed

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 556–565, 2010.
© Springer-Verlag Berlin Heidelberg 2010

an algorithm that calculates data summarization using summary functions and that performs appropriate transformations.

2 Summarization Process

In this paper, the process of XML (schema and data) summarization is divided into the following three stages (Figure 1):

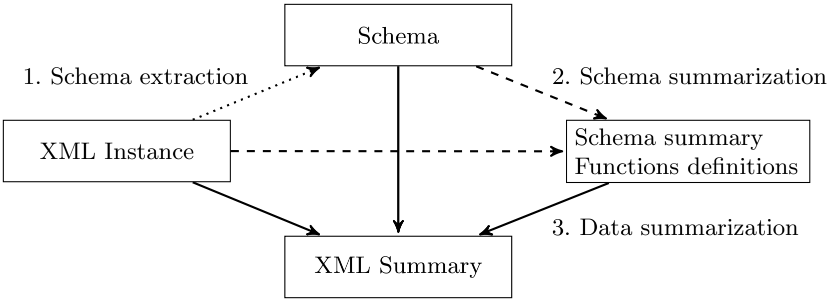


Fig. 1. Summarization process

1. **Schema extraction**. For a given XML document its XML schema must be extracted. As has been noted in [2] most XML data in the internet is available without a schema or with the schema of poor quality. Very rarely XSD schemas contains more information than can be written using simple DTD. Therefore, a more universal approach is to extract schema from the source data. We assume that information about keys of XML elements is discovered along with the structure of XML data.
2. **Schema summarization**. As the result of an analyse of the schema and its instance(s), the most significant nodes of the schema must be chosen to create schema summary of the required size (number of schema nodes). Then these nodes are used to create a schema that can be treated as a summary of the source schema.
3. **Data summarization**. In accordance to the relationship between the source schema and its summary, an instance of the schema summary must be obtained. So, an instance of the source schema must be transformed to form valid according to the summarized schema. During transformation mapping functions are used to create data summary, that can meet user expectations.

3 Schema Summarization

Schema summarization is a process in which we choose most representative nodes from a given schema and create another schema called its *summary*. The size of schema summary is determined by the number of its nodes. There is no optimal number of nodes to create a good summary. One user may want to get only a few nodes although they may not provide all vital information. Other may want to see more complicated summary, because he is already somehow familiar with the schema. We assume that the size of a summary is given, so it can be easily adjusted to meet user's needs.

To create an expected schema summary, we need information provided by a source schema and its instance(s).

Definition 1 (Schema). *A XML schema is a directed graph, $S=<\mathcal{L},\mathcal{E},\rho,\kappa>$, where: (a) $\mathcal{L}$ is a finite set of labels; (b) $\mathcal{E}$ is a finite set of directed edges between labels, $(l_1 \rightarrow l_2) \in \mathcal{E}$; (c) ρ is the root label, $\rho \in \mathcal{L}$; (d) κ is a function that, assigns keys to labels, $\kappa : \mathcal{L} \rightarrow 2^{\mathcal{L}}$.*

Definition 2 (Instance). *A XML instance is a labeled directed graph, $I =< N, E, r, \lambda, \nu >$, where: (a) N is a finite set of nodes; (b) E is a finite set of directed edges between nodes, $(n_1 \rightarrow n_2) \in E$, when n_1 is parent of n_2; (c) r is the root node, $r \in N$; (d) λ is a function that assigns labels from $\mathcal{L}$ to nodes (the label l, such that $l = \lambda(n)$, is the type of n); (e) ν is a function assigning text values to nodes.*

A key for a label l is a set of labels $(l_1, ..., l_m)$ such that for any tuple of values, $(\nu(n_1), ..., \nu(n_m))$, $\lambda(n_i) = l_i$, there exists at most one subtree of type l in any instance of the schema [1,6].

An instance I must satisfy structural constraints and key dependencies to be an instance of a schema S, i.e. $\lambda(r) = \rho$, and if $(n_1 \rightarrow n_2) \in E$, then $(\lambda(n_1) \rightarrow \lambda(n_2)) \in \mathcal{E}$.

Let the exchange format used by ECTS Course Catalogue `http://ects.wmid.amu.edu.pl` be our running example in this paper. It describes information about teaching units, degree programme and courses. Its schema originally consists of about 100 elements and attributes, so only a small part that could fit on page is presented in Figure 2. Also an instance that conforms to this schema is available.

3.1 Selecting Nodes

The most significant nodes in a given schema S are identified by the PageRank algorithm [4]. The algorithm assigns a weight to each edge in the schema. The weights must be normalized to ensure algorithm's convergence. This measure of importance of schema labels was proposed in [7] and is based on labels connectivity in the schema and nodes cardinality in the instance.

Definition 3. *A relative cardinality of an edge $l_j \rightarrow l_k$, $RC(l_j \rightarrow l_k)$, is an average number of nodes of type l_j connected to nodes of type l_k.*

Let S be a schema and I be an instance of S. Then $RC(l_j \rightarrow l_k), l_j, l_k \in \mathcal{L}$ is expressed as the number of edges outgoing from nodes of type l_j to nodes of type l_k, divided by the number of all edges outgoing from nodes of type l_j, i.e.

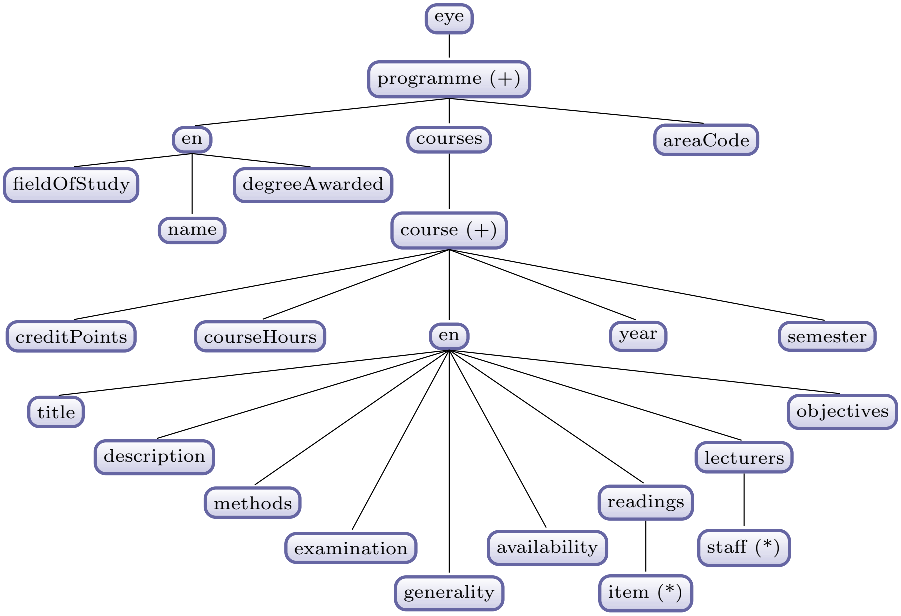


Fig. 2. ECTS Course Catalogue schema

$$RC(l_j \to l_k) = \frac{|\{(n_1, n_2) \in E : \lambda(n_1) = l_j \wedge \lambda(n_2) = l_k\}|}{|\{(n_1, n_2) \in E : \lambda(n_1) = l_j\}|}.$$

Definition 4. *Let S be a schema and I be an instance of S. The importance of $l \in \mathcal{L}$ with respect to I, written as $V_l(I)$, depends on its connectivity in S and its cardinality in I, and can be calculated with the following iterative formula until convergence is reached:*

$$V_l^q(I) = p \times V_l^{q-1}(I) + (1-p) \times \sum_j W_{l_j \to l} \times V_{l_j}^{q-1},$$

where: (a) $W_{l_j \to l} = \frac{RC(l_j \to l)}{\sum_k RC(l_j \to l_k)}$ is the neighbor weight, which is the relative weight of l from its parent label l_j, compared with all other children of l_j; (b) q denotes the number of iterations; (c) p, $0 \leq p \leq 1$, is a tuning parameter. The lower the p, the more the importance of a label is affected by the labels it is connected to; (d) the initial importance, $V_l^0(I)$, is set to the cardinality of the label in I.

With this formula, importance can be easily assigned to each schema label. Then k labels with highest importance can be chosen as candidates for the summary. In our running example (for K=10), the following labels with highest importance have been chosen: *eye*, *programme*, *courses*, *course*, *en*, *title*, *description*, *readings*, *item*, *lecturers* (root node is mandatory to create coherent schema).

3.2 Summary Graph Creation

Definition 5. *A summary for schema* $S =< \mathcal{L}, \mathcal{E}, \rho, \kappa >$ *is a pair* $(S', \mathcal{M})$*, where:*

- $S' =< \mathcal{L}', \mathcal{E}', \rho, \kappa' >$ *is a schema such that: (a)* $\mathcal{L}' \subseteq \mathcal{L}$*; (b) an edge* $(l \rightarrow l') \in \mathcal{E}'$*, if there is a connection between* l *and* l' *in* S*; (c)* S *and* S' *have the same root* ρ*; (d)* κ' *is defined only for* $l \in L'$*, that belonged to* $domain(\kappa)$ *and* $\kappa'(l) = \kappa(l) \cap L'$*;*
- $\mathcal{M}$ *is the representation relation and shows how labels from* S *are represented in* S'*, i.e.* $(l, l') \in \mathcal{M}$ *iff* $l' \in \mathcal{L}'$ *represents (summarizes, aggregates) the label* $l \in \mathcal{L}$*. Each label from* S *must have its representation in* S'*.*

Let K be a set of k most important labels of a schema S. Algorithm 1 creates a schema summary of S. To do this we use depth-first search algorithm. A label not belonging to the summary is removed, and its children are attached to its parent. The representation relation $\mathcal{M}$ is also updated accordingly to keep track of how labels are represented in the summary.

Algorithm 1. *Creation of summary for a schema*

```
Input:  $S =< \mathcal{L}, \mathcal{E}, \rho, \kappa >$ -  a schema,
        $K \subseteq \mathcal{L}$ - set of relevant labels for the summary;
Output: $(S' =< K, \mathcal{E}', \rho, \kappa' >, \mathcal{M})$ -  a summary of $S$.

function summarizeRoot()
   foreach $(node : (\rho \rightarrow node) \in \mathcal{E}')$
   summarizeNode(node, $\rho$)

function summarizeNode(node,parentNode)
   foreach $(childNode : (node \rightarrow childNode) \in \mathcal{E}')$
      summarizeNode(childNode,node)

   if $(node \in K)$
      $\mathcal{M} = \mathcal{M} \cup \{(node, node)\}$
   else
      $\mathcal{L}' = \mathcal{L}' \setminus \{(node)\}$
      $\mathcal{E}' = \mathcal{E}' \setminus \{(parentNode \rightarrow node)\}$
      $\mathcal{M} = \mathcal{M} \cup \{(node, parentNode)\}$
      foreach $(child : (node \rightarrow child) \in \mathcal{E}')$
         $\mathcal{E}' = \mathcal{E}' \cup \{(parentNode \rightarrow child)\}$
         $\mathcal{E}' = \mathcal{E}' \setminus \{(node \rightarrow child)\}$
Program:
$\mathcal{L}' = \mathcal{L}$
$\mathcal{E}' = \mathcal{E}$
$\mathcal{M} = \{\}$
summarizeRoot()
```

For schema on Figure 3 and nodes limit set to 10 the summary on Figure 3 is generated. The real source document was in fact much more complicated than presented schema, but on the summary you can easily see the general structure of the document.

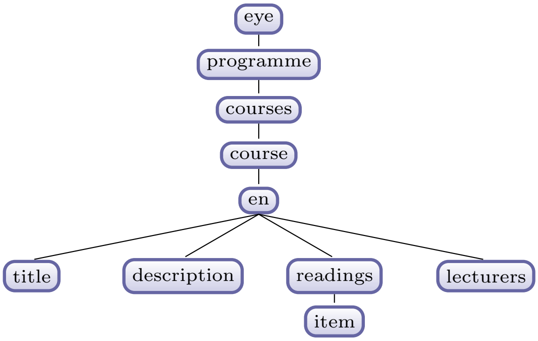


Fig. 3. ECTS Course Catalogue summary of size 10

4 Data Summarization

The purpose of this step is to present the data from source document in the form corresponding to the summary obtained in the previous step. It means that text values from an instance of a schema S must be summarized (aggregated) in the way consistent with the schema summary $(S', \mathcal{M})$.

4.1 Representation of Instances as Valuation Sets

We will represent an instance I of a schema S as a pair (S, Ω), where Ω is a set of *valuations* of text-valued labels of S.

Definition 6. *Let* $I =< N, E, r, \lambda, \nu >$ *be an instance of* $S = (\mathcal{L}, \mathcal{E}, \rho, \kappa)$*, and* $\mathcal{T} \subset \mathcal{L}$ *be a set of types of nodes in* N*, for which the function* ν *is defined, i.e.* $\mathcal{T}$ *is the set of text-valued labels (*$l \in \mathcal{T}$*, if there is a node* $n \in N$*, such that* $\lambda(n) = l$*, and* $\nu(n)$ *is defined). The valuation of* $\mathcal{T}$ *is a function* $\omega : \mathcal{T} \rightarrow Str \cup \{\perp\}$*, where* Str *is a set of string values, and* $\perp$ *is the empty (null) value.*

According to the definition above, we see that and any instance is a finite set of valuations of text-valued labels of its schema. Thus the set of valuations created from an instance may be treated as an alternative, flat representation of the instance document. Valuation set may be easily transformed back and an instance conforming to the schema can be obtained. In this way we can transform valuations to an instance of different schema, with similar data but different structure [5].

Example 1. A fragment from the set Ω of valuations of an instance of schema in Figure 2:

name	*areaCode*	*title*	*description*	...
Computer science	11.3	Mathematical analysis	Lecture of mathematical ...	...
Computer science	11.3	Introduction to algebra	Algebra is a science ...	...
Computer science	11.3	Programming languages 1	The course focuses on ...	...
Computer science	11.3	Discrete mathematics	This course deals with ...	...
...	...	...	...	...

The set of valuations for an instance is generated by algorithm 2.

Algorithm 2. *Algorithm generating valuation set from instance document*

```
Input:  $I =< N, E, r, \lambda, \nu >$ - instance document,
Output: $\Omega$ - valuation set for I.

function generateValuation(node)
   create empty valuation set: $\Omega \leftarrow \oslash$
   if $\nu(node) \neq \epsilon$ (node has not empty text value) then
      create valuation $\omega$, such that $\omega(type(node)) = \nu(node)$
      $\Omega \leftarrow \Omega \cup \{\omega\}$
   else
        foreach $(node' : (node, node') \in E)$
         $\Omega'$ = generateValuation(node')
         if $\Omega = \oslash$ then
            $\Omega \leftarrow \Omega'$
         else
            create empty valuation set: $\Omega'' \leftarrow \oslash$
            foreach $(\omega \in \Omega)$
               foreach $(\omega' \in \Omega')$
                  $\omega'' \leftarrow \omega \circ \omega'$
                  $\Omega'' \leftarrow \Omega'' \cup \{\omega''\}$
            $\Omega \leftarrow \Omega''$
   return $\Omega$
```

In Algorithm 2, $\omega'' \leftarrow \omega \circ \omega'$ denotes the function:

$$\omega''(l) = \begin{cases} \omega(l), & if\ l \in domain(\omega) \\ \omega'(l), & if\ l \in domain(\omega') \end{cases}$$

4.2 Summary Functions

Summary functions are used to compute a summary (an aggregated value) from values occurring in an instance I of a schema S (represented by a set Ω) into a summarized (aggregated) value being an instance I' of a summary schema S' (represented by a single valuation $\{\omega'\}$). A summary function is a superposition of two functions: a "horizontal" p-ary function f that combines values of a fixed number of labels (e.q. concatenation of values of *title* and *author*), and a "vertical" function φ defined over an arbitrary set of values (e.g. *sum, avg, min, max, count, representative, concatenation*).

Definition 7. *Let S be a schema with the set $\mathcal{T}$ of text-valued labels, and $(S', \mathcal{M})$ be its summary, and $\mathcal{T}'$ be the set of text-valued labels in S'. Let Ω be a set of valuations of $\mathcal{T}$. Then the valuation ω' of $\mathcal{T}'$ is an $\mathcal{M}$-summary of Ω, if: $\omega' = \{(l', v) : l' \in \mathcal{L}', v = \varphi_{l'}(f_{l'}(\omega(l_1), ..., \omega(l_p))), \omega \in \Omega, (l_i, l') \in \mathcal{M}\}$, where: (a) $f_{l'} : Str^p \rightarrow Str$ is a function over set of text values; (b) $\varphi_{l'} : Str^{|\Omega|} \rightarrow Str$ is an aggregate function.*

In general, it is not possible to automatically find all dependencies between arbitrary schemas, but in the case of summarization the problem is much simpler.

We only need to find mappings between nodes from source schema and their subset chosen for summary. We know also the representation relation $\mathcal{M}$. Such mappings can be divided into the following classes:

- identity mapping (one-to-one)
- functional dependency (one-to-one)
- functional dependency (many-to-one)
- functional dependency + aggregate function (many-to-one)

There is no need to define many-to-many mapping, because such a mapping wouldn't give us information how to evaluate function formulas during transformation.

For automatic mapping definition we used only first three classes, because it is generally not possible to guess mapping function expected by a user. However user can easily define custom functions f and φ for the mapping. In our implementation, we have chosen a simplified XPath expressions for function definition, which seems to be simple and natural solution.

Example 2 (Mapping functions). For our running example we can use the following summary functions:

- identity mapping (one-to-one):
  $(f = title, \varphi_e = Id)_{title}$
- functional dependency (one-to-one):
  $(f = substring(title, 10), \varphi = Id)_{title}$
- functional dependency (many-to-one):
  (concatenation of title and description)
  $(f = toUpperCase(title) + description, \varphi = Id)_{description}$
- functional dependency + aggregate function (many-to-one):
  (nodes counting)
  $(f = staff, \varphi = count)_{lecturers}$

4.3 Transformation

The final step is creation of a summarized XML document that is an instance of the summary schema. In this process a structural transformation must be carried out. This is performed by Algorithm 3.

Algorithm 3. *Generation of summarized document*

```
Input:
$\Omega$ - valuation set
$S' = (< K, \mathcal{E}, \rho, \kappa >, \mathcal{M})$ -  a schema summary
$I =< N, E, r, \lambda, \nu >$ - empty target instance document ($N = \{r\}$)

function generateDocument($\Omega$, $n$, $l$)
   $keys = \kappa(l)$ (key set for label $l$)
   if $keys = \oslash$ then
```

```
    foreach ($l' \ : \ (l,l') \ \in \ \mathcal{E}$)
      create node $n'$, such that: $\lambda(n') \ = \ l'$
      attach element $n'$ to its parent: $E \ = \ E \ \cup \ (n,n')$
      generateDocument($\Omega$, $l'$, $n'$)
  else
    while $\Omega \ \neq \ \oslash$
      find valuations $\Omega'$ that have same values for key labels
      $\Omega' \ = \ \{\omega'\}$, for any $\omega' \ \in \ \Omega$
      $\Omega \ = \ \Omega \setminus \Omega'$
      foreach ($\omega \ : \ \omega \ \in \ \Omega$)
        if $\forall \ key \ \in \ keys \ (\omega(key) \ = \ \omega'(key))$ then
          $\Omega' \ = \ \Omega' \ \cup \ \Omega'$
          $\Omega \ = \ \Omega \setminus \Omega'$

      create node $n'$, such that: $\lambda(n') \ = \ l'$
      attach element $n'$ to its parent: $E \ = \ E \ \cup \ (n,n')$

      if $l'$ is of complex type: $\exists \ l' \ ((l,l') \ \in \ \mathcal{E})$
        foreach ($l' \ : \ (l,l') \ \in \ \mathcal{E}$)
          generateDocument($\Omega'$, $l'$, $n'_i$)
      else ($l'$ is of simple type)
        $\omega' \ = \ \mathcal{M}\text{-}summary(\Omega')$
        assign value to node: $\nu(n'_i) \ = \ \omega'(l')$
```

Example data summary is presented below. The real advantage of presented solution is that user can easily change the data included in the summary by changing mapping functions or the schema summary.

Example 3 (Data summary)

```
<eye>
  <programme>
    <courses>
      <course>
        <en>
          <title>Discrete mathematics</title>
          <description>This course deals ...</description>
          <readings>
            <item>K.A. Ross, Ch.R.B. Wright, Matematyka dyskretna, PWN, Warszawa 1996</item>
            <item>V. Bryant, Aspekty kombinatoryki, WNT - Warszawa, 1997</item>
          </readings>
          <lecturers>Dr Tomasz Schoen Dr Edyta Szymanska </lecturers>
        </en>
      </course>
      <course>
        <en>
          <title>Advanced programming with Delphi</title>
          <description>The aim of the course is to ...</description>
          <readings>
            <item>A. Marciniak, Borland Delphi 5 Professional, Poznan 2000.</item>
          </readings>
          <lecturers>Prof. Dr hab. Andrzej Marciniak</lecturers>
        </en>
      </course>
      ...
    </courses>
  </programme>
</eye>
```

5 Conclusion

We discussed data summarization for XML in which we summarize both schema and the data. We believe that this approach facilitates understanding of very complex schemas, because user can see the schema together with the data he wants to analyse. We defined universal formalism for mapping functions that supports data transformation and data summarization. The user can easily choose which data he wants to include in the summary. We presented algorithms and their open source implementation as XTR system.

All mentioned algorithms are implemented within XTR that is available at `http://www.xtr.sf.net` [3]. Some technical aspect like attributes and mixed types elements were not mentioned in this paper, but we made effort to support standards as much as we could. Therefore algorithm in the implementation are a bit more complicated. Application is free and open source with the following main features:

- XML, XSD Summarization
- XML Transformation
- XML Merging
- XSD Schema extraction - xml2xsd
- XML Validation (Syntax validation, Validation with XML Schema - XSD)
- XML Edition - with auto formatting and syntax highlighting
- XSD visualization (using tex with tikz library).

Acknowledgement. The work was supported in part by the Polish Ministry of Science and Higher Education under Grant 3695/B/T02/2009/36.

References

1. Arenas, M., Libkin, L.: A normal form for XML documents. ACM Trans. Database Syst. 29, 195–232 (2004)
2. Bex, G.J., Neven, F., Schwentick, T., Tuyls, K.: Inference of Concise DTDs from XML Data. In: VLDB, pp. 115–126. ACM, New York (2006)
3. Marciniak, J., Pankowski, T.: Automatic XML data transformation and merging. Zeszyty Naukowe Wydzialu ETI Politechniki Gdańskiej. Technologie Informacyjne 16, 231–236 (2008)
4. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web, Technical report (1999)
5. Pankowski, T.: XML data integration in SixP2P – a theoretical framework. In: EDBT Workshop Data Management in P2P Systems (DAMAP 2008). ACM Digital Library, pp. 11–18 (2008)
6. Pankowski, T., Cybulka, J., Meissner, A.: XML Schema Mappings in the Presence of Key Constraints and Value Dependencies. In: ICDT 2007 Workshop EROW 2007. CEUR Workshop Proceedings, vol. 229, pp. 1–15. CEUR-WS.org (2007)
7. Yu, C., Jagadish, H.V.: Schema Summarization. In: VLDB, pp. 319–330. ACM, New York (2006)

Sample-Based Collection and Adjustment Algorithm for Metadata Extraction Parameter of Flexible Format Document

Toshiko Matsumoto, Mitsuharu Oba, and Takashi Onoyama

Research and Development Division, Hitachi Software Engineering Co., Ltd.
4-12-7, Higashishinagawa, Shinagawa-ku, Tokyo, 140-0002, Japan
{tmatsumoto,ohba,onoyama}@hitachisoft.jp

Abstract. We propose an algorithm for automatically generating metadata extraction parameters. It first enumerates candidates on the basis of metadata occurrence in training documents, and then examines these candidates to avoid side effects and to maximize effectiveness. This two-stage approach enables both avoidance of exponential explosion of computation and detailed optimization. An experiment on Japanese business documents shows that an automatically generated parameter enables metadata extraction as accurately as a manually adjusted one.

Keywords: logical structure analysis, metadata extraction, keyword extraction, layout characteristics.

1 Introduction

In recent years, companies have created and stored more and more documents in digital form. This increase in the number of digital documents has heightened two kinds of needs. First, business documents must be carefully handled and protected against information leakage. Second, accumulated documents can be used to improve business efficiency. Enterprise Content Management (ECM) systems are expected to address these needs. They manage business documents using metadata such as titles, creation dates, and customer names.

Methods of metadata extraction from document content (in other words, logical structure analysis) have been proposed for science articles or fixed-form documents [1], [2], and have been studied for business documents [3], [4]. Previous methods require a prior setting of an intended document area such as articles or fixed-form documents, detailed investigation of describing of metadata in such an area, and definition of the description as a "parameter" for metadata extraction. We call this parameter "metadata extraction (ME) parameter". Table 1 shows an example of an ME parameter. For each kind of metadata, strings are extracted when the strings appear next to a neighbor keyword, contain a substring keyword, or have a maximum score as a weighted sum of with or without layout characteristics. Not only the preceding but also the following neighbor keyword is effective (for example, "(including this sheet)" can be an effective following neighbor keyword for total page number in a facsimile coversheet).

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 566–573, 2010.
© Springer-Verlag Berlin Heidelberg 2010

Table 1. Example of Metadata Extraction parameter

information in a ME parameter	kinds of metadata	
	Title	Customer name
neighbor keyword	"Subject: "	"To: " or "Billing address"
substring keyword	"Notification"	"Co., Ltd."
weight vector	"center alignment" * 2 + "bold face" * 1	"close to upper-left hand corner" * 1

Because of this metadata extraction architecture, its accuracy depends on the quality of the ME parameter; however, preparing a well adjusted ME parameter for each document area is laborious. Moreover, the inefficiency of manual adjustment can prevent a metadata extraction method from being applicable to business. There are algorithms that automatically induce layout characteristics for each kind of metadata [5], [6]. However, they are for documents where each kind of metadata has a fixed number of lines. Because of this limitation, they cannot be applied to flexible format business documents created with office software such as MS Word or Excel. Algorithms have been proposed for parameter adjustment for parsing scientific bibliographic reference [7]. However, their input data is a plain-text, while a two-dimensional location of characters is important in business document interpretation.

We propose the "Sample-based Collection and Adjustment (SCA) algorithm", which uses sample documents and their manually specified metadata as training data, and generates an ME parameter for metadata extraction. We evaluate our algorithm by comparing the accuracy of metadata extraction with an automatically generated ME parameter to that with a manually adjusted ME parameter.

In Section 2, we describe problems in calculating an ME parameter. In Section 3, we propose our SCA algorithm and describe its calculation process in detail. Finally, experimental results and discussion are given in Sections 4 and 5, respectively.

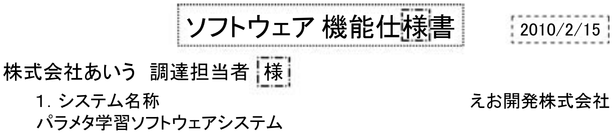
2 Problems in Calculating a ME Parameter

There are three problems in calculating an ME parameter. First, automatically generated ME parameter has to enable metadata extraction as accurately as a manual adjusted ME parameter does. Second, our SCA algorithm has to avoid following side effects. Third, our algorithm has to search for an effective parameter within practical calculation time.

There are two types of side effects that may occur with an ME parameter. The first type pertains to keywords. Inappropriately selected keywords may appear without metadata. For example, a neighbor keyword of a customer's name appears in the title, as shown in Fig. 1 (enclosed with dashed-dotted line); therefore, the prefix of the title can be misinterpreted as a customer name. Ideographic characters in Japanese and Chinese seem to make this side effect be apparent.

However, this side effect actually occurs with every language. Strings with unintended occurrence are not suitable for keywords. The second type pertains to layout characteristics. For example, “underline” is intuitively expected to contribute to title extraction because sometimes titles are underlined. However, ID numbers and prices are underlined more frequently. Therefore, a weight vector, which attaches too much importance to “underline”, is not suitable for distinguishing titles.

The search space of an ME parameter is large because of the arbitrariness of keywords and the wide variety of layout characteristics. Each metadatum can have any string as a keyword, and there is no number limitation. Furthermore, several tens of layout characteristics can be used (examples are listed in Table 2). Therefore, our SCA algorithm must efficiently search for an effective parameter.



ソフトウェア 機能仕様書 2010/2/15

株式会社あいう 調達担当者 様

1. システム名称

パラメタ学習ソフトウェアシステム

えお開発株式会社

Fig. 1. Sample function specification document of software. Creation date (*enclosed with dashed line*) is next to title (*enclosed with dotted line*). Characters second from right in title and neighbor keyword of customer’s name are same (*enclosed with dashed-dotted lines*).

Table 2. Examples of layout characteristics

No.	Layout characteristics	No.	Layout characteristics
1	Bold face	10	Close to lower left-hand corner
2	Center alignment	11	Close to lower right-hand corner
3	Underline	12	Close to upper left-hand corner
4	Enclosed with line	13	Close to upper right-hand corner
5	Absence of vertical neighbor string	14	Vertical small font
6	Vertical wide-spaced	15	Vertical large font
7	Minor font on page	16	Large font
8	Minor font color on page	17	Upper half of page
9	Minor background color on page	18	Smallest sized font on page

3 Algorithm for Calculating a ME Parameter

We discuss the concept of our SCA algorithm and describe its calculation process in detail. To solve the three problems described in Section 2, we developed our SCA algorithm with the following three features. First, it examines occurrence of manually specified metadata in training documents to collect information required to extract the metadata. Neighbor keywords are collected from strings close to the metadata, and substring keywords are collected from substrings of metadata. Layout characteristics have large weight values when the characteristics are observed, particularly in training document metadata. Second, our algorithm prevents the above-mentioned side effects. To examine whether each keyword and layout characteristic is specific to a kind of metadata, non-metadata strings and other kinds of metadata strings are used. Third, our algorithm generates an ME parameter in two-stages: collecting candidates and detailed adjustment. This two-stage approach enables our algorithm to optimize keywords and a weight vector without exponential explosion of computation.

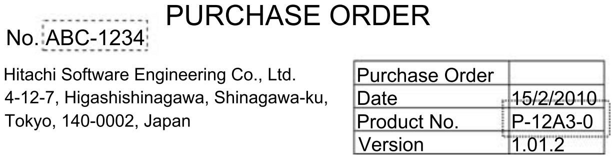
Next, we describe in detail how our SCA algorithm calculates neighbor keywords, substring keywords, and a weight vector of layout characteristics. Calculation of keywords and a weight vector are explained in Sections 3.1 and 3.2, respectively.

3.1 Calculating Neighbor and Substring Keywords

From each occurrence of metadata in training documents, one or more neighbor strings of the metadata are collected for neighbor keyword candidates, and substrings of the metadata are collected for substring keyword candidates.

To filter out inconsistent or inefficient candidates, our SCA algorithm examines occurrences of keyword candidates in each training document according to two rules. First, a keyword candidate must be of a kind of metadata. Fig. 1 shows an example where one string is of a neighbor keyword candidate of one kind of metadata and of a substring keyword candidate of another kind. The creation date (enclosed with dashed line) is next to the title (enclosed with dotted line) and therefore, creation date can be falsely collected as a neighbor keyword candidate of the title. With this rule, the creation date will not be a neighbor keyword of the title. Fig. 2 shows another example where a neighbor keyword candidate of one kind of metadata and a neighbor keyword candidate of another kind share the same string. Suppose that a string "No." is collected as a neighbor keyword candidate of an order form number (enclosed by dashed line). However, "No" is contained in a string next to the product number (enclosed by dotted line). This rule prevents "No" from extracting the product number as an order form number.

Second, a keyword candidate must not be substantially less frequent. When more than five training documents contain a kind of metadata and a keyword candidate is collected from only one training document, the keyword



No. ABC-1234

PURCHASE ORDER

Hitachi Software Engineering Co., Ltd.
4-12-7, Higashishinagawa, Shinagawa-ku,
Tokyo, 140-0002, Japan

Purchase Order	
Date	15/2/2010
Product No.	P-12A3-0
Version	1.01.2

Fig. 2. Sample document contains order form and product numbers. Their neighbor keyword candidates are enclosed with dashed and dotted lines, respectively.

candidate is discarded. This rule can filter out accidentally collected strings, which is extremely less frequently and not effective in extracting metadata from other documents. The threshold value of five was determined from a preliminary experiment.

3.2 Calculating Weight Vector of Layout Characteristics

Our SCA algorithm first evaluates each layout characteristic to select ones that would be effective for extracting metadata. Then a weight vector is optimized.

Our SCA algorithm evaluates how each layout characteristic can be effective in extracting metadata. The effectiveness of “center alignment” for extracting a title is described to explain the evaluation process. For each training document, the title and other strings are examined to determine if they are center aligned. When the title is center aligned and all the other strings are not, the title can be extracted with “center alignment” for the document. In contrast, when the title is not center aligned and some other strings are center aligned, the title is not extracted with “center alignment”. The effectiveness of “center alignment” is evaluated by the difference between the number of training documents whose titles can be extracted and the number of training documents whose titles cannot be extracted. All layout characteristics are evaluated in this way. Up to four layout characteristics are selected according to their effectiveness as long as the layout characteristic extracts the title from one or more training documents. In our experiment, described in Section 4, three layout characteristics were selected at most. Therefore, the threshold value of four is considered to be large enough.

To successfully calculate the score for metadata extraction, the weight vector is adjusted for the selected layout characteristics. Changing the weight value from 0 to 3 for each layout characteristic, the weight vector is optimized to maximize the difference between the number of training documents, whose metadata can be extracted by the weighted sum score, and number of training documents, whose metadata cannot be extracted. The optimized weight vector is adopted when the difference value is positive. Otherwise, the kind of metadata is considered to be unsuitable for layout characteristic-based extraction.

4 Experimental Results

We evaluated our SCA algorithm with Japanese documents from six business deals (Table 3). Our algorithm was compared with a manually adjusting process according to the following five steps: The business deals were divided into two groups of three deals (step 1), our SCA algorithm generated an ME parameter from the group-1 documents (step 2), metadata was extracted from the group-2 documents with the ME parameter generated in step 2 (step 3), steps 1 to 3 were repeated with all different division patterns (step 4), and the accuracy rate with a manually adjusted ME parameter was calculated from all six business deals and was compared with the averaged accuracy rate from step 4 (step 5).

Table 3. Business documents used in evaluation. Second to eighth columns show number of documents where kind of metadata is written.

	Number of documents						
Deal No.	Title	Creation date	Customer name	Estimate ID number	Order entry ID number	Contract number	Order form number
1st	12	7	11	4	2	3	4
2nd	27	11	20	5	0	11	0
3rd	13	5	9	1	0	2	0
4th	15	10	12	2	0	2	2
5th	35	10	11	3	0	3	0
6th	29	16	20	5	0	4	3

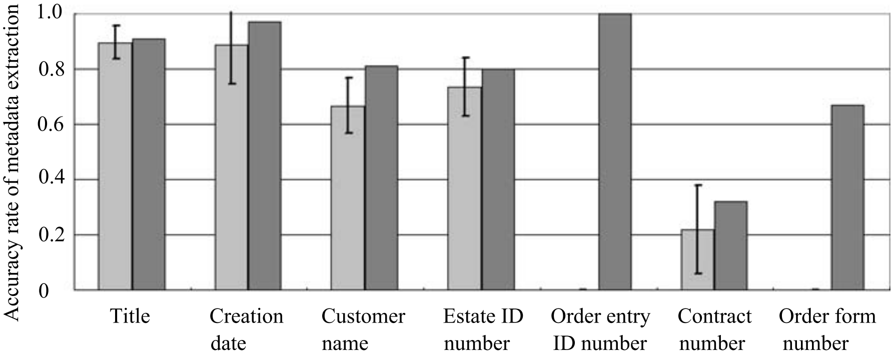


Fig. 3. Accuracy rate of metadata extraction with automatically generated (*light grey*) and manually adjusted (*dark grey*) ME parameters. Vertical bars are standard deviations.

Dividing business deals (not dividing documents) in step 1 closely aligns the experimental results with an actual introduction of the ECM system to a company. Certain kinds of metadata, such as customer name, estimate ID number, or order form number may have identical values in all the documents of one business deal. With this dividing policy, metadata values are definitely different between training and testing datasets. Fig. 3 shows the accuracy rates of metadata extraction with manually adjusted and automatically generated ME parameters. On average, our SCA algorithm took 28 seconds on a PC with a Core 2 Duo (2 GHz) CPU and 3 GB of memory.

5 Discussion

Our SCA algorithm solved all three problems described in Section 2. For the first problem, Fig. 3 shows that an automatically generated ME parameter enabled metadata extraction as a accurately as manually adjusted one for five of seven kinds of metadata. For the other two kinds, order entry ID and order form numbers, only a few documents were available as a training dataset, as shown in Table 3. Therefore, they should be excluded from evaluation. Our SCA algorithm also solves the second problem. In narrowing down keyword candidates, our SCA algorithm eliminates candidates that may cause side effects. With this process, the SCA algorithm enables extensive keyword collection and prevents inconsistency of collection. In selecting layout characteristics, our SCA algorithm evaluates the side effects for each layout characteristic with observation of the characteristics in non-metadata strings. Therefore, our SCA algorithm uses only promising layout characteristics. Since the computation time was less than 1 minute, our SCA algorithm solves the third problem. The two-stage approach helps our SCA algorithm focus on promising candidates, even though a keyword can be arbitrarily selected and several tens of layout characteristics can be used.

The importance of the number and variety of training data is indicated as follows. Assuming frequencies of keywords are approximated with a Zipfian distribution [8], probabilities of collecting all keywords from training documents is calculated, as shown in Table 4. From this table, we can say that 50 occurrences are almost always ($\geq$ 90%) enough to collect all keywords, even when as many as five keywords are required. In fact, the number is expected to be large enough for more than five keywords because less frequent keywords do not contribute as much to metadata extraction. Our experimental results present moderate correlation (correlation coefficient 0.68) between the accuracy rate of metadata extraction and the number of metadata occurrences in training data, and they almost always shows a good accuracy rate ($\geq$ 80% with a few exceptions) when more than 50 occurrences are available for calculating an ME parameter of a certain kind of metadata. The exceptions indicate the importance of a variety of training documents. For extracting customer names from Japanese business documents, two neighbor keywords are effective. Since documents in first, second, and sixth business deals lacked one of the neighbor keywords, our SCA algorithm could not collect it when these three deals were grouped as training data. For this reason, the accuracy rate of customer name extraction was not good in spite of a sufficient number of occurrences in the training data.

We proposed the SCA algorithm, which generates an ME parameter for metadata extraction from sample documents and their metadata as a training data. It solves all three problems in calculating a ME parameter for metadata extraction for flexible format documents: accurate metadata extraction, avoidance of side effects, and a practical calculation time. Furthermore, we presented

Table 4. Probabilities of collecting all keywords from training documents

	Number of occurrences in training documents								
Number of keywords	10	15	20	25	30	35	40	45	50
3	0.42	0.74	0.89	0.95	0.98	0.99	1.00	1.00	1.00
4	0.07	0.32	0.57	0.75	0.85	0.92	0.95	0.97	0.99
5	0.00	0.07	0.24	0.43	0.59	0.72	0.81	0.87	0.92

a rough indication of the required number of training data. Our SCA algorithm enables a metadata extraction method to be applied to a wide variety of document areas with manageable labor costs. We believe that our algorithm can be effective not only for Japanese business documents but also for English documents because metadata extraction from English documents requires neighbor keyword and substring keywords (as listed in Table 3, Fig. 1 and 2) and layout characteristics [2]. On the other hand, additional enhancement can make our SCA algorithm more effective for English documents (e.g., stemming or dehyphenating words). From an algorithmic viewpoint, our SCA algorithm uses rather simple calculation, and treats each kind of metadata separately. It can treat all kinds of metadata simultaneously through formalizing ME parameters as association rules [9]. Moreover, our SCA algorithm can be more effective in the operation phase of metadata extraction with a function of improving a precomputed ME parameter by adding a sample document.

References

1. Taylor, S.L., Fritzson, R., Pastor, J.A.: Extraction of Data from Preprinted Forms. Machine Vision and Applications 5, 211–222 (1992)
2. Lee, K., Choy, Y., Cho, S.: Geometric Structure Analysis of Document Images: A Knowledge-Based Approach. IEEE Trans. on PAMI 22, 1224–1240 (2000)
3. Minagawa, A., Fujii, Y., Takebe, H., Fujimoto, K.: A Method of Logical Structure Analysis for Form Images with Various Layouts by Belief Propagation. IEIC Technical Report 106, 17–22 (2006)
4. Ishitani, Y.: Logical Structure Analysis of Document Images Based on Emergent Computation. In: 5th International Conference on Document Analysis and Recognition, pp. 189–192 (1999)
5. Esposito, F., Malerba, D., Semeraro, G., Ferilli, S., Altamura, O., Basile, T.M.A., Berardi, M., Ceci, M., Di Mauro, N.: Machine Learning Methods for Automatically Processing Historical Documents: from Paper Acquisition to XML Transformation. In: 1st International Workshop on Document Image Analysis for Libraries, pp. 328–335 (2004)
6. Wnek, J.: Machine learning of generalized document templates for data extraction. In: Lopresti, D.P., Hu, J., Kashi, R.S. (eds.) DAS 2002. LNCS, vol. 2423, pp. 457–468. Springer, Heidelberg (2002)
7. Kramer, M., Kaprykowsky, H., Keysers, D., Breuel, T.: Bibliographic Meta-Data Extraction Using Probabilistic Finite State Transducers. In: 9th International Conference on Document Analysis and Recognition, pp. 609–613 (2007)
8. Zipf, G.K.: Selected Studies of the Principle of Relative Frequency in Language. Cambridge (1932)
9. Mitra, S., Acharya, T.: Data Mining: Multimedia, Soft Computing and Bioinformatics. Wiley-Interscience, Hoboken (2003)

A New Stochastic Algorithm for Strategy Optimisation in Bayesian Influence Diagrams

Michał Matuszak and Tomasz Schreiber

Faculty of Mathematics and Computer Science,
Nicolaus Copernicus University,
Toruń, Poland
{gruby,tomeks}@mat.umk.pl

Abstract. The problem of solving general Bayesian influence diagrams is well known to be NP-complete, whence looking for efficient approximate stochastic techniques yielding suboptimal solutions in reasonable time is well justified. The purpose of this paper is to propose a new stochastic algorithm for strategy optimisation in Bayesian influence diagrams. The underlying idea is an extension of that presented in [2] by Chen who developed a self-annealing algorithm for optimal tour generation in traveling salesman problems (TSP). Our algorithm generates optimal decision strategies by iterative self-annealing reinforced search procedure, gradually acquiring new information while driven by information already acquired. The effectiveness of our method has been tested on computer-generated examples.

1 Introduction

Influence diagrams[4,5,6] are widely acknowledged as an important probabilistically oriented graphical representation paradigm for decision problems. An influence diagram is built on a directed acyclic graph (DAG) whose nodes and arcs admit standard interpretations steming from and extending those used for Bayesian (belief) networks. Three principal types of nodes are considered: chance nodes standing for random variables (represented as ovals in our figures below), decision nodes corresponding to available decisions (rectangles in our figures) and utility nodes (rhombi) specifying the utilities to be maximized by suitable choices of decision policies. The arcs leading to chance nodes indicate direct causal relationships (at least if the network is well designed) not necessarily corresponding to any temporal ordering. On the other hand, the arcs leading to decision nodes specify the information available at the moment of decision making, thus feeding input to decision policies. Some arcs between decision nodes may also be of informative nature as determining the order of decision making. The influence diagrams can be considered as a generalization of (symmetric) decision trees, see [4].

In Fig. 1, generated by Hugin Lite package [3], a simple example of an influence diagram is shown. The decision node *treatment* represents the choice whether or not to visit a doctor. A visit to a doctor does increase the chance of no cough, yet

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 574–581, 2010.
© Springer-Verlag Berlin Heidelberg 2010

it also causes negative effects, such as the need to pay the fee for visit. Further, wearing a *scarf* decreases the chance of getting sore throat but also negatively affects our appearence. All these effects are jointly taken into account in the utility node *happiness*.

A number of algorithms for solving influence diagrams have been developed, falling beyond the scope of the present article. We refer the reader to Chapter 10 in [4] for a detailed discussion, see also the references in Subsection 5.2.2 of [5].

In general, finding an optimal decision strategy for an influence diagram is an NP-hard task. This is easily shown by reducing an NP-complete problem to the considered task. A natural choice is the traveling salesman problem (TSP) known as a classical NP-complete task. To each city a decision node is ascribed with the remaining cities as admissible states. The decision taken coincides with the next city to visit. Further, a utility node is created with incoming arcs from all decision nodes. The utility function is defined by summing up the negative distances between cities and their successors in case the decision sequence yields a valid Hamilton tour, and is set to $-\infty$ otherwise. Clearly, solving the TSP problem is, in this set-up, equivalent to finding the optimal decision strategy for the so-constructed influence diagram.

In his work [2] (see also the discussion in [7]) Chen proposed an appealing and simple stochastic optimisation algorithm for the TSP problem, quite original in its design, highly effective and yet apparently somewhat underestimated in the literature. In the course of an iterative procedure subsequent TSP tours are randomly generated: each city is assigned a table of weights for connections to all remaining cities and each time the choice of the next city to visit is made by random among cities not yet visited, with probabilities proportional to the corresponding connection weights. This way all cities get visited and eventually we get back to the starting point. Next, the so generated tour is compared with the one obtained in the previous iteration. Depending on whether the new cycle is longer or shorter than the previous one, the connection weights between cities neighbouring in both tours are correspondingly reinforced or faded. The algorithm stops when the re-normalised weights are close enough to zeros and ones, which corresponds to a deterministic tour choice, converging to the optimal one under suitable reinforcement/fading protocols.

With the TSP problem regarded as a particular case of decision strategy optimisation, the purpose of the present paper is to is to extend Chen's approach to general influence diagrams. As we will see, this can be done neatly and effectively, although not without substantial extensions of Chen's idea.

2 The Algorithm

To give a formal description of the proposed decision strategy optimisation algorithm, assume an influence diagram $(\mathcal{S}, \mathcal{P}, \mathcal{U})$ is given, built on a connected DAG $\mathcal{S}$, with CPTs $\mathcal{P}$ and utility functions $\mathcal{U}$. The set of nodes in $\mathcal{S}$ splits into chance nodes $C_{\mathcal{S}}$, decision nodes $D_{\mathcal{S}}$ and utility nodes $U_{\mathcal{S}}$. All these objects are

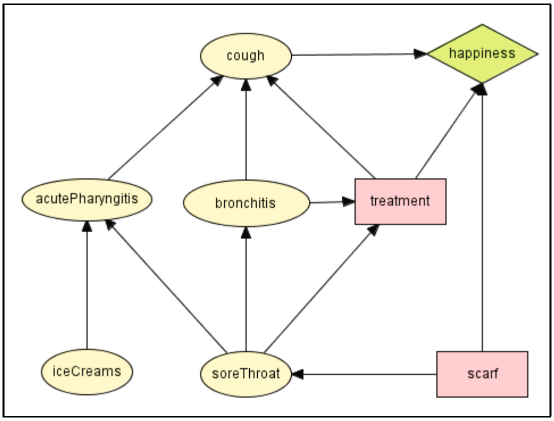


Fig. 1. Sample influence diagram

assumed fixed in the course of strategy optimisation. In addition, for optmisation purposes we attach to each decision node $D \in D_{\mathcal{S}}$ the *randomised policy* δ_D which assigns to each configuration $\bar{w}$ of $\mathrm{pa}(D)$ a probability distribution on possible decisions to be taken, that is to say $\delta_D(d|\bar{w})$ stands for the probability of choosing decision d given that $\mathrm{pa}(D) = \bar{w}$. These randomised policies will evolve in the course of the optimisation process, eventually to become (sub)optimal deterministic policies collectively determining the utility maximizing strategy for the influence diagram considered. The initial choice of $\delta_D,\ D \in D_{\mathcal{C}}$ can be either *uniform*, with all decisions equiprobable, or *heuristic* provided some additional knowledge is available on our system allowing us to make a good *first guess* about the optimal strategy.

Our algorithm relies on an iterative procedure carried out in *epochs* of fixed length N. At the beginning of each epoch $t,\ t = 1, 2, \ldots$ we divide the collection $D_{\mathcal{C}}$ of decision nodes into the set of *trainable decision nodes* $TD^t_{\mathcal{C}}$ whose randomised decision policies are to undergo updates, and *frozen decision nodes* $FD^t_{\mathcal{C}}$ whose status is to remain unaltered throughout the epoch. Roughly speaking, the purpose of this splitting is to ensure that only a modest fraction of system parameters are updated at a time, which is indispensible for the stability of algorithm, see below for a more detailed discussion. In addition, we impose the following requirement, whose relevance is discussed in the sequel.

[**Forbidden path condition.**] No two trainable nodes are connected by a directed path in $\mathcal{S}$.

With $\mathcal{D} \subseteq D_{\mathcal{S}}$ write $A[\mathcal{D}]$ for the ancestry of $\mathcal{D}$ in $\mathcal{S}$ that is to say $A[\mathcal{D}]$ is the collection of nodes in $\mathcal{S}$ from which a decision node from $\mathcal{D}$ can be reached along a directed path in $\mathcal{S}$. Clearly, the forbidden path condition is equivalent to requiring that $A[TD^t_{\mathcal{S}}] \cap TD^t_{\mathcal{S}} = \emptyset$ during tth epoch. Moreover, since the utility nodes have no progeny, we readily conclude that $A[TD^t_{\mathcal{S}}] \subseteq C_{\mathcal{S}} \cup FD^t_{\mathcal{S}}$. A standard way of selecting the trainable collection, as implemented in our software, goes as follows.

- Whenever passing to a new epoch, do sequentially for all decision nodes $D \in D_{\mathcal{C}}$
  - If D is a trainable node then freeze it with some probability p_F,
  - If D is a frozen node, make it trainable with some probability $p_T < p_F$ unless doing so violates the forbidden path condition and unless the fraction of time during which D was trainable exceeds maximal admissible value (specified as algorithm parameter).

Note that the *quota* imposed on the fraction of time a given node is trainable is aimed at preventing the situation where a decision node with numerous progeny in $D_{\mathcal{C}}$ receives only a very poor training time fraction as predominantly blocked by its progeny.

The following iterative procedure, repeated a fixed number N of times during each optimisation epoch, say tth epoch, and directly motivated by the ideas developed in [2], lies at the very heart of our algorithm.

1. Set the iteration counter $i := 0$.
2. Generate an instance $\bar{w}_A^{(i)}$ of the *ancestral variable configuration* for $A[TD^t_{\mathcal{S}}] \subseteq C_{\mathcal{S}} \cup FD^t_{\mathcal{S}}$ according to the CPTs of chance nodes and using the randomised decision policies of frozen decision nodes in $FD^t_{\mathcal{S}} \cap A[TD^t_{\mathcal{S}}]$ as CPTs. This is carried out in the standard way with nodes handled recursively proceeding from causes to effects in the policy subnetwork $A[TD^t_{\mathcal{S}}]$. This is where the forbidden path condition is of use as ensuring that no trainable node falls into $A[TD^t_{\mathcal{S}}]$.
3. Repeat a fixed number M of times
   (a) Sample the decisions taken, $d_1, \ldots, d_k$, independently for all trainable decision nodes $D_1, \ldots, D_k \in TD^t_{\mathcal{S}}$ according to their respective current randomised policies $\delta_{D_j},\ j = 1, \ldots, k$ given $\bar{w}_A^{(i)}$. Note that $\bigcup_{j=1}^{k} \mathrm{pa}(D_j) \subseteq A[TD^t_{\mathcal{S}}]$ and thus the knowledge of $\bar{w}_A^{(i)}$ is sufficient for this sampling.
   (b) Evaluate the expected total utility

$$u^{(i)} := \mathbb{E}[U_{\text{total}} | \bar{w}_A^{(i)}; d_1, \ldots, d_k]$$

   given $\bar{w}_A^{(i)}$ and $d_1, \ldots, d_k$, under the randomised policies of frozen nodes used as respective CPTs. This is easily done by standard Monte-Carlo network instance generating scheme, recursively proceeding from causes to effects. This is possible because the non-instantiated part of the network $\mathcal{S} \setminus [A[TD^t_{\mathcal{S}}] \cup TD^t_{\mathcal{S}}]$ contains no trainable decision nodes and it has the upward cone property – whenever it contains a node X it also contains all its children and, inductively, its whole progeny.
   (c) If $i \geq 1$, set

$$\mathit{\Delta} := u^{(i)} - u^{(i-1)}$$

   and update the policies $\delta_{D_j},\ j = 1, \ldots, k$ for trainable nodes by putting

$$\delta_{D_j}(d_j | \bar{w}_A^{(i)} \cap \mathrm{pa}(D_j)) := \exp(\beta \mathit{\Delta}) \delta_{D_j}(d_j | \bar{w}_A^{(i)} \cap \mathrm{pa}(D_j))$$

and, thereupon, re-normalising $\delta_{D_j}(\cdot|\bar{w}_A^{(i)} \cap \mathrm{pa}(D_j))$ so that it remain a probability distribution. The positive constant β here is a parameter of the algorithm, the larger it is the faster the system learns but the less stable the optimisation is.

4. Set $i := i + 1$ and, if $i < N$, return to 1. Otherwise terminate the current epoch.

The intuitive meaning of the above procedure is that the network is presented with a configuration sampled according to the CPTs and current randomised policies of the frozen nodes, whereupon the randomised policies of the trainable nodes are used for decision sampling, with succesful choices (positive Δ) leading to reinforcement of the corresponding probability entries and, on the other hand, with choices deteriorating the performance resulting in fading of the corresponding probability entries (negative Δ). The reinforcement/fading strength depends on the value of the utility gain/loss compared to the previous run. In analogy to Chen's work [2], also here after a large enough number of epochs we eventually end up with the situation where all randomised policies become nearly deterministic in that, for each $D \in D_{\mathcal{C}}$ and parent configuration $\bar{w}$ for $\mathrm{pa}(D)$ the value of $\delta_D(d|\bar{w})$ is close to one for a unique d and close to zero otherwise. This determinism can be easily quantified by looking at the maximal value of $\min(\delta_D(d|\bar{w}), 1 - \delta_D(d|\bar{w}))$ and declaring a policy *nearly deterministic* when this falls below, say, 0.01. To sum up, our algorithm carries out subsequent optimisation epochs until all policies become nearly deterministic. Note in this context that is is crucial to ensure that each node is trainable during a sufficiently large fraction of time, for otherwise it might long remain untrained slowing down the entire process. As already mentioned, this is handled by our training selection scheme discussed above.

3 Implementation and Examples

The programme has been implemented in language `D` [8], currently gaining popularity as a natural successor to `C++`, and uses the Tango library [1]. The implementation, aimed so far mainly at algorithm evaluation purposes, can be described as careful but not fully performance-optimised, with the total utility evaluation under frozen decision nodes in 3.(b) performed using the standard Monte-Carlo rather than a more refined and effective scheme. The graph of the diagram was represented using neighbourhood lists. The utility functions were

Table 1. Decision policy for node *treatment*

Y	N	
1.00	0.00	sore throat = Y, bronchitis = Y
0.99	0.01	sore throat = Y, bronchitis = N
0.03	0.97	sore throat = N, bronchitis = Y
0.01	0.99	sore throat = N, bronchitis = N

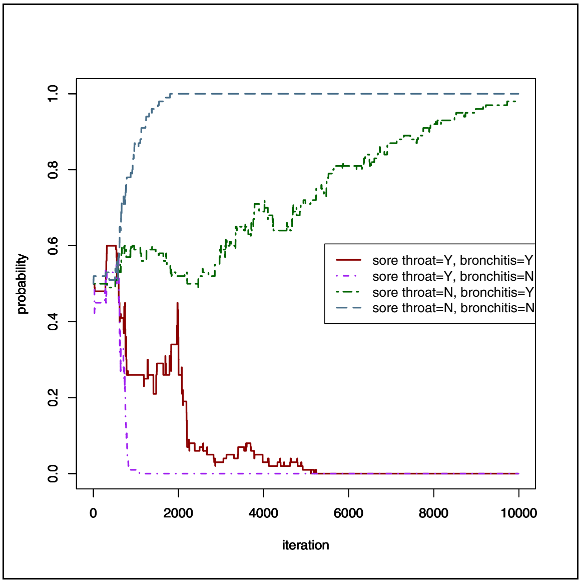


Fig. 2. Convergence of decision policies

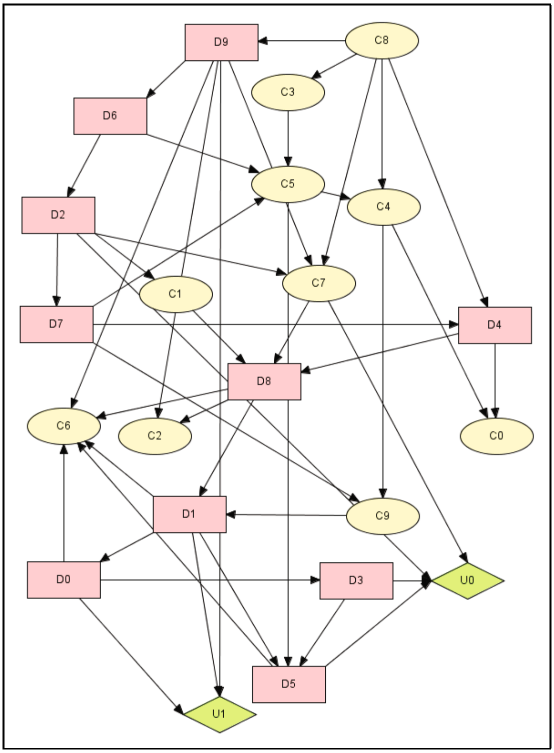


Fig. 3. Randomly generated influence diagram

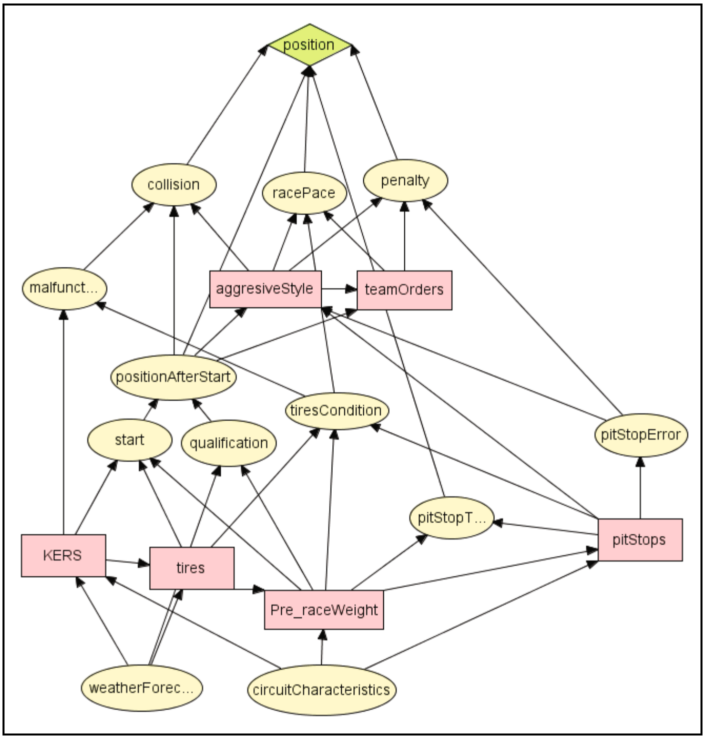


Fig. 4. Race results

always fit to $[0, 1]$. All test runs were executed on a machine with Intel Core 2 Q9300 2.50 GHz CPU and 2GB RAM.

A sequential version of our algorithm run for a randomly generated influence diagram with 10 chance nodes, 10 decision nodes, 10 utility nodes and 40 arcs, see Fig. 2 generated from Hugin Lite [3], required 1.1ms time per epoch. The algorithm parameters were set as follows: $p_T = 0.01$, $p_F = 0.04$ and $\beta = 0.01$. After 10000 epochs (11 seconds) the strategy output by our algorithm achieved utility only 4% inferior to the optimal one (as determined using the Hugin package). For a different randomly generated network (doubled number of nodes) execution time per one epoch was 1.8ms, with 10000 epochs. Again, the utility of the output strategy was only 4% inferior to the optimal one.

Getting back to the network depicted in Fig. 1, we performed 10000 iterations of our algorithm, with parameters $p_T = 0.01$, $p_F = 0.04$ and $\beta = 0.03$. The convergence of decision policies for node *treatment* in the course of our algorithm is shown in Fig. 2. The table 2 represents the policy obtained upon convergence of the algorithm. It can be concluded that the sore throat is of crucial importance for our decision whether or not to visit a doctor. On the other hand, the bronchitis appears to be ignored. This does coincide with the optimal strategy as determined by the Hugin Lite package.

For the diagram given in Fig. 3 (12 chance nodes, 6 decision nodes, one utility node and 42 arcs) the mean execution time per epoch was 0.8ms. The decision strategy after 10000 iterations with parameters $p_T = 0.01$, $p_F = 0.04$ and $\beta = 0.01$ was inferior by 5 % to the optimal strategy.

References

1. Bell, K., Igesund, L.I., Kelly, S., Parker, M.: Learn to Tango with D. Apress (2008)
2. Chen, K.: Simple learning algorithm for the traveling salesman problem. Phys. Rev. E 55, 7809–7812 (1997)
3. http://www.hugin.com
4. Jensen, F.V., Nielsen, T.D.: Bayesian Networks and Decision Graphs, 2nd edn. Springer, Heidelberg (2007)
5. Neapolitan, R.E.: Learning Bayesian Networks. Prentice Hall Series in Artificial Intelligence. Pearson Prentice Hall, London (2004)
6. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers Inc., San Francisco (1988)
7. Peretto, P.: An Introduction to the Modeling of Neural Networks. In: Collection Aléa-Saclay. Cambridge University Press, Cambridge (1992)
8. http://www.digitalmars.com/d/

Forecasting in a Multi-skill Call Centre

David Millán-Ruiz[1], Jorge Pacheco[1], J. Ignacio Hidalgo[2], and José L. Vélez[1]

[1] Telefónica Research & Development, Emilio Vargas, 6, 28043 Madrid, Spain
[2] Complutense University of Madrid, 28040 Madrid, Spain
dmr@tid.es, jorge.pacheco@altran.es, hidalgo@dacya.ucm.es, jlvv@tid.es

Abstract. Call centre technology is subject to multiple improvements and innovations. Some of them try to improve agent performance and client satisfaction. There are two different but intrinsically linked problems. The first one is related to predictions of call arrivals, call abandonment, available agents having a certain skill, call pick-up and average delay time; and the second one deals with workload distribution within a multi-skill Call Centre. In this paper, we focus on forecasting call arrivals, which can be approached from several angles. Specifically, we analyse and compare Neural Networks, Time Series and Regression Models in this study.

Keywords: Time Series, Neural Networks, Regression, Forecasting.

1 Introduction

A Call Centre (CC) is a centralised place used for receiving and transmitting large volumes of requests by telephone [1]. In a CC, the flow of calls is often divided into outgoing and incoming traffic. *Incoming calls* are those that go from the client to the CC to contract a service, ask for information or report a problem. These calls are modelled and classified into several call groups (CGs) in relation to the nature of each call. Once these CGs have been modelled, each call is assigned to a unique CG. Conversely, *outgoing calls* are significantly dissimilar to incoming calls since these calls are made by the agents with commercial pretensions.

Predictions are vital in making management decisions in CCs to properly distribute workload among agents. A fair distribution of workforce improves client satisfaction and reduces costs. In this way, an initial step to produce a planning is to predict future system loads, comprising both predicted arrivals and mean service times. Intuitively, the mean arrival rate for each CG is not the same and their calls involve different modelling times. Note that incoming flow in CCs is usually not a stationary Poisson process [2] and, the service times do not increase exponentially. Since calls randomly arrive according to a stochastic process, a very balanced distribution of the agents is needed in order to handle the calls as soon as possible. For this reason, our main concern is to identify the best method to forecast incoming call arrivals.

The rest of this paper is organised as follows: Section 2 summarises the state of the art on forecasting in CCs. Section 3 presents a comparative study on forecasting techniques for multi-skill CCs. In Section 4, the conclusions of this study are given.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 582–589, 2010.
© Springer-Verlag Berlin Heidelberg 2010

2 Previous Work on Forecasting in CCs

Forecasting refers to the estimation of output values in unknown situations. In our domain, a precise prediction enables us to properly balance the workload among agents, giving higher service levels and, eventually, optimising our resources.

This section is organised as follows: Subsection 2.1 addresses the classical Poisson distribution. Subsection 2.2 briefly exposes a common regressing technique. Subsection 2.3 presents some Time Series methods. Subsection 2.4 covers this problem from Neural Networks' point of view.

2.1 Poisson Distribution

Traditionally, forecasting in CCs has been approached by a Poisson Distribution (PD). Assuming pure-chance arrivals and pure-chance terminations leads to the following probability distribution (1):

$$P(n) = \left(\frac{\mu^n}{n!} \right) e^{-\mu} \tag{1}$$

PD expresses the probability of a number of events occurring within a timeslice, when these events are independent of the previous event and take place with a known average. Under these conditions, it is a reasonable approximation of the exact binomial distribution of events. PD provides a useful mechanism to assessing the percentage of time when a given range of results will be expected. In the calculation of the distribution function, the values for the mean and standard deviation are carried over from the binomial distribution.

In this way, "conventional" approaches assume that the number of call arrivals at a given time, t, follows a PD, where n is the number of call arrivals in an interval of duration d, and μ is the mean of call arrivals at time t. For this reason, pure-chance traffic is also known as Poisson traffic. However, as previously mentioned, the prediction of call arrivals in a CC does not often follow a PD with a deterministic rate [2]. In all studies, the arrival process agrees with a Poisson process only if the arrival rate of the Poisson process is itself a stochastic process. Characteristically, the variance of the incoming calls in a given interval is much larger than the mean whereas it should be equal to the mean for PDs. The mean arrival rate also strongly depends on the day time and often on the week-day. Finally, there is positive stochastic dependence between arrival rates in successive periods within a day and arrival volumes of successive days. Next sections describe other procedures to forecast unknown variables, taking into account the nature of a real CC environment.

2.2 Regression Model

A Regression Model (RM) [3] is a statistical method in which an unknown variable is predicted according to its relation with the rest of well-known variables, using a formula called *regression equation*. This equation deals with some constant parameters which must be optimised in order to reduce the Mean Square Error (MSE) between the predicted output and its real value. In particular, we study *Lineal Regression* (LR) which is the commonest method. LR approximates the unknown variable with a straight line by using well-known variables as follows:

$$Y_i = \beta_0 + \sum \beta_p X_{ip} + \varepsilon_i \tag{2}$$

where parameter i is the pattern-position in the dataset, p is the n-th well-known variable, β_P represents the associated parameters to the n-th well-known variable, β_0 is a constant parameter, Y is a dependent variable and ε is the associated error. β_P and β_0 are calculated in order to reduce $\Sigma\varepsilon_i$, using well-known patterns.

The main advantage of this method is the clearness to understand it. But, it is hard to choose the variables to generate the model, considering seasonality and trend, which is crucial to better understand the behaviour of a CC [4].

2.3 Time Series

Time Series (TS) is a sequence of observed variables in regular timeslices. This sequence is used to understand and forecast the behaviour of a given variable over the time [5]. A TS approximates a future value by applying a regression (more or less complex) on the n-previous variables to estimate forthcoming values. TS can be divided into two major groups: Exponential Smoothing (ES) and Autoregressive Integrated Moving Average (ARIMA). ES assigns decreasing weights to each previous observation. At the same time, ES methods are divided into: Simple TS, Dumped Trend Time Series and Stationary Time Series.
Simple TS or Single Exponential Smoothing [6] can be calculated as follows:

$$S_t = \alpha y_{t-1} + (1-\alpha)S_{t-1}, \quad 0 < \alpha \leq 1, t \geq 3; \qquad F_{t+1} = \alpha y_t + (1-\alpha)S_t, \tag{3}$$

Dumped Trend Time Series or Holt's Linear Model [6] can be calculated as:

$$\begin{aligned} &S_t = \alpha y_{t-1} + (1-\alpha)(S_{t-1} + b_{t-1}), \quad 0 \leq \alpha \leq 1 \\ &b_t = \gamma(S_t - S_{t-1}) + (1-\alpha)b_{t-1}, \quad 0 \leq \gamma \leq 1; \qquad F_{t+m} = S_t + mb_t \end{aligned} \tag{4}$$

Stationary Time Series or Holt-Winters' Trend and Seasonality Model [6] can be expressed as follows:

$$\begin{aligned} &S_t = \alpha \frac{y_t}{I_{t-L}} + (1-\alpha)(S_{t-1} + b_{t-1}), \quad 0 \leq \alpha \leq 1; \qquad b_t = \gamma(S_t - S_{t-1}) + (1-\gamma)b_{t-1}, \quad 0 \leq \gamma \leq 1 \\ &I_t = \beta \frac{y_t}{S_t} + (1-\beta)I_{t-L}; \qquad F_{t+m} = (S_t + mb_t)I_{t-L+m} \end{aligned} \tag{5}$$

where y is the observation, S is the smoothed observation, b is the trend factor, I is the seasonal index, F is the forecast at m periods ahead, t is an index which denotes a time period and α, β and γ are constants which must be estimated. In this way, the MSE is tried to be minimised. But, the main advantage of Exponential Smoothing TS is that it requires short computing times [7]. Nevertheless, the model cannot accurately predict for a long timeslice [4]. To mitigate this handicap, we generate a daily model to forecast the following day (explained in Section 3.1).

Differently, ARIMA [6] is determined by three parameters *(p, d, q)*, where *p* is the autoregressive term, *d* is the number of previous values and *q* is the average moving parameter. ARIMA (p, d, q) can be calculated for a TS sequence Y_t *(t=1,2,..., n)*, as follows:

$$\begin{aligned} &\phi(B)(1-B)^d Y_t = \theta(B)Z_t; \quad \textit{where} \ ,\phi(B) = (1-\alpha_1 B - \alpha_2 B^2 - \ldots - \alpha_p B^p) \\ &\textit{and}\ \theta(B) = (1-\beta_1 B - \beta_2 B^2 - \ldots - \beta_q B^q) \qquad (6) \\ &\textit{and}\ \mathrm{Z}_{\mathrm{t}}\ \text{is a white noise sequence and B is the backshift operator} \end{aligned}$$

ARIMA (p,q,d)·(P,D,Q) represents a multiplication of two ARIMAs to include seasonality to the model. This method needs that new seasonal and non-seasonal parameters are estimated, analogously to simple ARIMA. The involved parameters are the following ones: *p* is the AR order which indicates the number of parameters of *φ*, *d* is the number of times that data series must be differenced to induce a stationary series, *q* is the MA order which indicates the number of parameters of *θ*, *P* is the seasonal AR order which indicates the number of parameters of *φ*, *D* is the seasonal MA order which indicates the number of parameters of *θ*, and *Q* is the number of times that a data series needs to be differenced to induce a seasonal stationary series.

The principal advantage of ARIMA TS is that it usually suites better than Exponential Smoothing TS, although this model requires long computing times [8] and poorly forecast for large time-horizons [4,9]. To mitigate this handicap, we generate daily models to forecast the forthcoming day as explained in Section 3.1.

2.4 Artificial Neural Networks

Basically, an artificial neural network (ANN) is a mathematical model based on the operation of biological NNs [10]. This model can be used as an effective method to forecast. A neuron is the simplest processing element of an ANN. Neurons are organised into two units: the first one adds products of weight coefficients and input signals and the second one follows a nonlinear function, commonly known as neuron activation function. The function, *f*, sums the weights and maps the results to an output, *o*. Neurons can be divided into three types of layers: *input*, *hidden* and *output*. The input layer is composed by neurons that represent the data input variables and "feed" next layers of neurons. Next layers are denominated hidden layers and there may be several of them. The last layer is called output layer, in which each neuron represents an output variable. Each layer is fully connected to the succeeding layer.

On the one hand, the main advantage of ANNs is their flexibility to make patterns, being suitable for large and complex datasets as well as long-time-horizon forecasting [8,11]. On the other hand, we can also find some disadvantages. These are the following ones: long computing times, risk of overfitting (see Section 3.1), need of a feature selection process and difficulty to approach all parameters for each CG [9]. In this paper, we consider the ANN described in [13] which claims promising results.

3 Comparative Study

In this section, we study the behaviour of the previously described techniques to choose the most appropriate one, considering our huge volume of data. This decision considers the quality of the results obtained according to the training time required.

3.1 Data Source

Firstly, a suitable dataset must be selected, looking for a fair balance between the amount of data and a representative period of time measured in terms of days. In this way, the number of selected days must be a multiple of *seven* because the predictor "week-day" has an important influence. Moreover, the choice of the number of days must be large enough to represent every possible pattern. However, this training-set should not be immense because this might drastically decrease the performance of the training process. Therefore, the number of selected days should be, at least, *91* days in order to cover all possible patterns with the previously exposed considerations.

Our problem presents *321* CGs, hence, the dataset is too large to make a complete study for all of them due to the elevated computing-time charge. Consequently, *five* CGs (CG1 is the most difficult one whereas CG5 is the easiest one) with different behaviour in terms of oscillations, arrival rates, processing times and nature, have been carefully picked in order to have a generic enough approach. Then, a different model has been developed for each CG. In this way, the whole dataset has been split into subsets considering each CG. Another choice is, due to the poor accuracy of TS for large horizon predictions, to calculate a model to forecast each following day with the daily dataset during the last two weeks of the dataset. In contrast, only one model is calculated to forecast the last two weeks for ANN and Regression. In the model generation, the last two weeks of each dataset have been used as validation-set, which has not been previously shown to the model, to verify the quality of each technique. The rest of the dataset has been randomly divided into two groups: training-set (60%) and generalisation-set (40%) to implement a cross-validation structure [12]. For TS, where a generalisation set is not used, all the dataset has been used as training-set. The training dataset is used to make the model learn the patterns whereas the generalisation dataset is used to prevent it from overtraining.

3.2 Feature Selection

Feature selection is the broadly applied technique of selecting a subset of relevant features or predictors in order to build robust learning models. As stated along these lines, our main concern is to predict incoming calls. Intuitively, the number of incoming calls is a probabilistic process influenced by many factors or variables. Choosing the right inputs from all information we have, is not trivial and is very important for obtaining a higher performance. Since these variables cannot be taken ad-hoc, the Mann–Whitney–Wilcoxon (MWW) test has been used to obtain a metric of the relevance of the variables (see Table 1). MWW test is a non-parametric test for calculating whether two independent samples of observations have the same distribution. This test has been chosen because it is one of the best-known non-parametric significance ones.

Table 1. Chosen-variable relevance according to MWW

Variables	Relevance %
# Calls in Previous 0-5 Minutes	41,171%
# Calls in Previous 5-10 Minutes	17,857%
Night Shift Timetable	11,741%
Week day	8,41%
# Calls in Previous 10-15 Minutes	6,307%
# Calls in Previous 15-20 Minutes	4,873%
# Calls in Previous 20-25 Minutes	4,790%
Minutes of the Day	3,433%
Peak Time	1,415%
Second Peak Time	1,398%

Additionally, to prevent us from taking wrong decisions, a study of influence has been done. This study reveals how the volume of incoming calls is influenced by variables such as week-day, day-time, peak hours, campaigns, festivities, night-shift timetable or volume of incoming calls in previous intervals (see Table 1). Other variables may also influence the results in some CGs sometimes but not significantly enough for most of them. Since a quick time response is required, these variables have not been considered.

3.3 Metrics

Metrics are usually specific for a given subject area and are often valid only within a certain domain and cannot be directly interpreted outside it. To measure the quality of each result within our domain, two traditional forecasting metrics have been selected: Mean Absolute Error (MAE) and Standard Deviation (SD). In order to make the process more understandable, the error has been defined as the difference between the real and predicted values. A model is better when MAE and SD are closer to 0.

3.4 Analysis of Results

Observing the variables which are more significant for incoming call forecasting (see Section 3.2), different methods have been analysed for 5-minute time-intervals with the aim of choosing the most suitable one for *this problem*. In this way, Figure 1 and Table 2 illustrate the MAE comparative between TS, ARIMA, RM and ANN. These confirm that, although each CG has a different behaviour and requires a different model to obtain the best approximation, the ANN regularly better behaves. ANNs present better results when CGs follow a complex pattern, e.g. CG2 and CG3.

Besides, while ARIMA and TS emphasise the "recent past"; ANNs are more flexible [9,11]. Table 2 demonstrates that the SD obtained by the ANN is also lower for all CGs, except for CG1 which is similar. This means that the error is similar to the mean error, making the prediction more reliable and stable for all intervals.

ANN also presents stable behaviour for all CGs as Figure 1 shows. CGs with few calls, i.e. CG5, present higher performance when the threshold is equal to five calls, reaching a level of accuracy up to 100%. Reasonably, those groups with huge volumes of incoming calls have higher absolute errors; while the error, in terms of percentage, is alike (see Figure 1). For this reason, other authors use a percentage dependent on the size to measure the quality of the prediction. Passed some epochs, the ANN learns more slowly because it is harder to minimise the error since there are many surrounding local optimums. The risk of overfitting due to the fast learning is covered with the generalisation dataset as other traditional methods do. This strategy is the correct one to optimise the computing times, which is a critical factor in this problem, allowing us to reach solutions close to the global optimum in a few epochs.

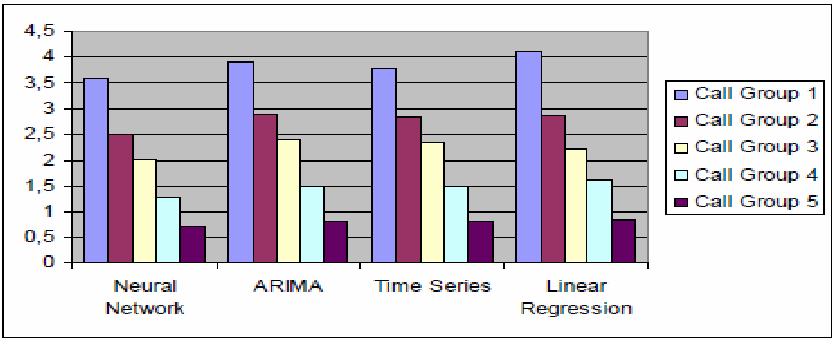


Fig. 1. General Comparison (MAE x Model)

Table 2. MAE & SD Comparison

Call Group	ANN MAE	ANN SD	ARIMA MAE	ARIMA SD	TS MAE	TS SD	RM MAE	RM SD
Group 1	3,591	6,381	3,892	6,430	3,791	6,360	4,12	6,501
Group 2	2,502	4,472	2,872	4,851	2,841	4,821	2,87	4,507
Group 3	2,012	3,542	2,378	4,087	2,349	4,104	2,21	3,551
Group 4	1,387	2,682	1,494	2,785	1,486	2,771	1,62	2,718
Group 5	0,718	1,205	0,823	1,408	0,819	1,406	0,84	1,313

Finally, AM and TS present accurate results for CGs with stable behaviour and/or low incoming flow but, compared to ANN, the results are still poor for CGs with complex behaviour, i.e. CG2 and CG3. The ANN models usually offer better results than TS for CGs with a large volume of calls, i.e. CG1 and CG2.

4 Conclusions and Future Work

We have addressed the difficulty of forecasting in real-world applications and, particularly, seen the importance of predicting in making management decisions within a multi-cast call centre. Afterwards, we have done a literature survey on forecasting, explaining some techniques which are being widely applied on similar problems by many researches and engineers. Then, we have presented the particularities of our call centre and the feature selection process that we have done. After that, we have proposed some metrics with the intention of comparing the different approaches. Finally, we have compared and analysed these techniques among them in order to choose the most suitable one.

We have also seen that the best forecast method for *our problem* is the ANN because it offers the tiniest MAE and SD. The reason is that, while ARIMA and TS techniques focus on the *recent past*, ANNs are more flexible for longer time horizons.

Eventually, we think that future work should focus on more sophisticated NN learning algorithms which might speedup the process, giving lower MAEs.

Acknowledgements

This work has been partially supported by Spanish Government grants TIN2008-00508 and MEC Consolider Ingenio CSD00C-07-20811 of the Spanish Council of Science and Technology. The authors would also like to thank Severino F. Galan for his comments and support on this work.

References

1. Bhulaii, S., Koole, G., Pot, A.: Simple Methods for Shift Scheduling in Multiskill Call Centers. M & SOM 10(3), 411–420 (Summer 2008)
2. Ahrens, J.H., Ulrich, D.: Computer Methods for Sampling from Gamma, Beta, Poisson and Binomial Distributions. Computing 12(3), 223–246
3. Heji, C., De Boer, P.M.C., Franses, P.H., Kloek, T., Van Dijk, H.K.: Econometric Methods with Applications in Business and Economics. Oxford University Press, Oxford (2004)
4. van den Bergh, K.: Predicting Call Arrivals in Call Centers, `http://www.few.vu.nl/stagebureau/werkstuk/werkstukken/werkstuk-bergh.pdf`
5. Brockwell, P., Davis, R.A.: Time Series: Theory and Methods. Springer, New York (1991)
6. Makridakis, S., Wheelwright, S.C., Hyndman, R.J.: Forecasting: Methods and Applications. John Wiley and Sons, Inc., Chichester (1998)
7. Antipov, A., Meade, N.: Forecasting call frequency at a financial services call centre. Journal of the Operational Research Society 53(9), 953–960 (2002)
8. Cotez, P., Rio, M., Rocha, M., Sousa, P.: Internet Traffic Forecasting using Neural Networks. In: International Joint Conference on Neural Networks (2006)
9. Zaiyong, T., Fishwick, P.A.: Feed-forward Neural Nets as Models for Time Series Forecasting. ORSA Journal of Computing (1993)
10. Mandic, D., Chambers, J.: Recurrent Neural Networks for Prediction: Architectures. In: Learning algorithms and Stability. Wiley, Chichester (2001)
11. Shabari, A.B.: Comparision of Time Series Forecasting Methods Using Neural Networks and Box-Jenkins Model. Matematika 17(1), ISSN 01278274
12. Black, J., Benke, G., Smith, K., Fritschi, L.: Artificial Neural Networks and Job-specific Modules to Assess Occupational Exposure. British Occupational Hygiene Society 48(7), 595–600 (2004)
13. Pacheco, J., Millán-Ruiz, D., Vélez, J.L.: Neural Networks for Forecasting in a Multi-skill Call Centre. In: EANN 2009, London, UK, August 27-29 (2009)

Identification of Load Parameters for an Elastic-Plastic Beam Basing on Dynamic Characteristics Changes

Bartosz Miller, Zenon Waszczyszyn, and Leonard Ziemiański

Rzeszów University of Technology,
ul. W. Pola 2, 35-959 Rzeszów, Poland
{bartosz.miller,zewasz,ziele}@prz.edu.pl
http://www.prz.edu.pl

Abstract. Single load parameters are identified on the base of changes of known dynamic characteristics of an elastic-plastic steel beam. It is also loaded by a control load in order not to involve characteristics of the initial structure. Special attention is paid to the location of measurement points to obtain accuracy of computations corresponding to possibilities of planned measurement devices. Finite Element Method was used for the simulation of dynamic characteristics and Standard Neural Networks were applied for the inverse analysis. The main goal of the paper is the formulation of a new non-destructive method in the area of health monitoring of civil engineering structures.

Keywords: Identification, Finite Element Method (FEM), Standard Neural Network (SNN), Semi Bayesian Neural Network (SBNN), control load, control measurement point.

1 Introduction

Damage or yielding of material in structures can lead to decrease of their load-carrying capacity [1]. In order to prevent a dangerous state of the structure the Structure Health Monitoring (SHM) analysis is performed. From among many of SHM methods the modal analysis is frequently applied, supported on measurement of dynamic responses. Then theses responses are used as input data in the identification analysis [2].

In the paper, a hybrid approach with two components (FEM, SNN) is used, where FEM is the Finite Element Method and SNNs are Standard (Deterministic) Neural Networks. The modal approach can easily be applied to the identification analysis in case of elastic structures, since the structure dynamic characteristics are constant. Local damage or yielding of material cause coupling of input variables, see [3], and mechanical characteristics depend on deformation of structures [4]. In the hybrid system, FEM is used for the simulation (forward) analysis to prepare patterns suitable for the learning of SNN. The trained SNN can be applied as an efficient tool in the identification (inverse) analysis. The

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 590–597, 2010.
© Springer-Verlag Berlin Heidelberg 2010

hybrid system FEM&SNN is of low degree of fusing since the FEM and SNN are used separately, applying the 'off line' computational technique, see [4].

The present paper is a continuation of [5], where a simple supported elastic-plastic beam was investigated. Two load parameters were identified, i.e. the resultant load R and its location l, cf. Fig. 1a. In order to tune the values of structure dynamic characteristics, i.e. eigenvalues f_j and eigenvectors v_j, an additional control load K is applied to the beam at the distance l_K, see Fig. 1a. Due to such an approach, discussed in [6], the changes of dynamic characteristics can be referred to the tuned structures. In such a way the dynamic parameters of the initial (undamaged) structure can be omitted in the identification analysis.

The location of control points, in which accelerations are measured (or quasi measurements are simulated by FEM), was fixed in [5]. In the present paper, optimized locations of measurement points are searched in order to minimize the network approximation errors. The dynamic tuning of the beam enables trimming its frequencies above the frequency band Δf (in the paper $\Delta f = 0.25\,Hz$ is assumed), related to measurement possibilities of accessible accelerometers.

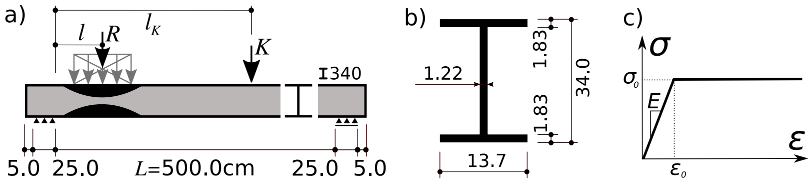


Fig. 1. a) Investigated steel beam, b) Dimensions of beam cross-section, c) Assumed material model

2 Beam and Material Models, Input and Output Data, FEM Simulations

The investigated beam is shown in Fig. 1a. The beam is made of I340 of cross-section of dimensions indicated in Fig. 1b. The mechanical characteristics correspond to a high-strength steel 18G2A, i.e. $E = 205\,GPa, \nu = 0.3, \sigma_0 = 345\,MPa$. The assumed elastic perfect plastic model of material is shown in Fig. 1c.

The FE simulation analysis was carried out by means of the commercial FE code ADINA [7]. The rectangular, 9-node, plain stress FE was used. Altogether, the beam was modelled by 3520 FEs. The input parameters correspond to geometrical and load data shown in Fig. 1a. The uniform load q was applied at ten different widths $w \in [5, 50]\,cm$. The load q was adopted for five levels r to give the resultant $R = q \cdot w \in [2.25, 14.50] \times 10^4\,N$. Twenty locations $l \in [20, 480]\,cm$ of the resultant load were considered. Control load corresponding to the fixed weight $K = 0.34 \times 10^4\,N$ was adopted, located at seven distances l_K. Altogether, the number of FE simulations is $10 \times 5 \times 20 \times 7 = 7000$. After preliminary computations it was stated that 198 simulations had to be eliminated since no initial yielding occurred in any beam cross-sections. Thus, $N = 6802$ patterns were adopted for the simulation analysis.

The geometrical and load parameters discussed above are used in the analysis of two eigen problems: a) only *main load* R is applied to the beam, b) both the main and a *control load* K are applied. The vectors of parameters $\mathbf{x}$ can be treated as an input vector of the forward analysis. This means that for the known values of parameters x_i dynamic parameters of the beam can be computed, analyzing the following eigen problems to find the eigenvalues f_j and eigenvector components v_{jP} in control points P:

$$\text{a)}\quad \mathbf{x}_{(2\times1)} = \{l, R\} \quad \rightarrow \omega_{(1\times5J)} = \{f_j, v_{jP}\}\,, \tag{1}$$

$$\text{b)}\ {}^c\mathbf{x}_{(3\times1)} = \{l, R, l_K\} \ \rightarrow\ {}^c\omega_{(1\times5J)} = \{{}^cf_j, {}^cv_{jP}\}\,, \tag{2}$$

where: $l = 1, \ldots, 20$; $R = 1, \ldots, 50$; $j = 1, \ldots, J$; $P = A, B, C, D$. The superscript c denotes the application of both the main and control loads to the beam.

3 Application of SNN in the Identification Analysis

Following the paper [5] the input vector ω' is composed of location of the control load l_K, changes of eigenfrequencies $\mathit{\Delta} f_j$ and eigenvectors components v_{jP}:

$$\omega'_{(1\times(1+5J))} = \{l_K, \mathit{\Delta} f_i, v_{jP}\}\,. \tag{3}$$

The changes of eigenfrequencies $\mathit{\Delta} f_j$ are defined as dimensionless relative increments:

$$\mathit{\Delta} f_j = \left| \frac{f_j - {}^cf_j}{f_j} \right| \,. \tag{4}$$

The output vector has two components corresponding to the identified load parameters:

$$\mathbf{y}_{(2\times1)} = \{l, R\}\,. \tag{5}$$

The data set is composed of the input/output pairs:

$$\mathbf{D} = \{\omega'_n, \mathbf{t}_n\}_{n=1}^{N=6802}\,, \tag{6}$$

where: $\mathbf{t}_n$ — target vector corresponding to known components of vector (5).

The data set (6) is randomly split into the learning (training), validation and testing sets, of pattern numbers $L = 0.6 \cdot N$, $V = 0.2 \cdot N$ and $T = 0.2 \cdot N$, respectively, where $N = 6802$ is the number of data set components.

The SNN formulation is based on the Last-Square-Error (LSE) measure:

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{\mathbf{t}_n - \mathbf{y}_n(\mathbf{x}_n; \mathbf{w})\}^2\,, \tag{7}$$

where: $\mathbf{w}$ — vector of network weights.

A more refined formulation, cf. Bayesian NNs [8,9], can be supported on the penalized LSE function, in which the barrier function $E_w(w)$ is added:

$$\hat{E}(\mathbf{w}) = \beta E(\mathbf{w}) + \alpha E_w(\mathbf{w}) = \frac{\beta}{2}\sum_{n=1}^{N}\{\mathbf{t}_n - \mathbf{y}_n(\mathbf{x}_n;\mathbf{w})\}^2 + \frac{\alpha}{2}\mathbf{w}^T\mathbf{w}\,, \qquad (8)$$

where: α, β — hyperparameters.

In the present paper feed-forward, layered SNNs were applied, see [9,10]. Single layer networks are composed of H tangential sigmoid functions and linear outputs. These SNNs were learnt by means of the Levenberg-Marquardt method. In case of the number of adopted eigenfrequencies $J = 4$ and for four control measurement points the SNN of structure 16-H-2 was adopted. After the application of the classical cross-validation method the optimal number of hidden neurons $H_{opt} = 28$ was evaluated.

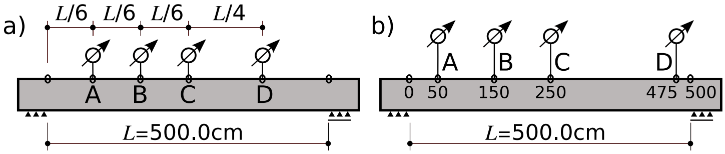


Fig. 2. Distribution of control measurement points: a) according to [2], b) after application of approximate optimization method discussed in Section 4

In [2] a family of Semi Bayesian NN were applied. These networks are based on error measure (8) and the hyperparameters are learnt by a Bayesian method, see [5]. In [2] a fixed distribution of the control points was applied, see Fig. 2a.

4 Optimized Location of Control Points

In the present paper an optimized (quasi optimal) distribution of control measurement points is introduced, assuming four points as assumed in [5]. The analysis of a corresponding optimization problem has to be carried out in 4D space of location parameters l_{Pi} for $i = 1, 2, 3, 4$. An error of NN approximation can be used as a cost function. The searching of the global minimum occurs to be very time consuming since in the problem considered a sequence of identification problems has to be solved. That is why an approximate method was formulated, similar to the classical Gauss-Seidel optimization method.

The computation starts from the searching of an optimal location of the first location point l_{P1}^{opt}. Then fixing the location l_{P1}^{opt} the second location l_{P2}^{opt} is evaluated. In the third step the computed locations are fixed and the third location l_{P3}^{opt} is searched. A value of l_{P4}^{opt} is proceeded at three fixed locations l_{P1}^{opt}, l_{P2}^{opt} and l_{P3}^{opt}. After a cycle of four steps the evaluated locations can be called *optimized* (quasi optimal) since the search of a global minimum has not been completed. Nevertheless, the four step method occurs to be numerically very efficient in the investigated identification problem.

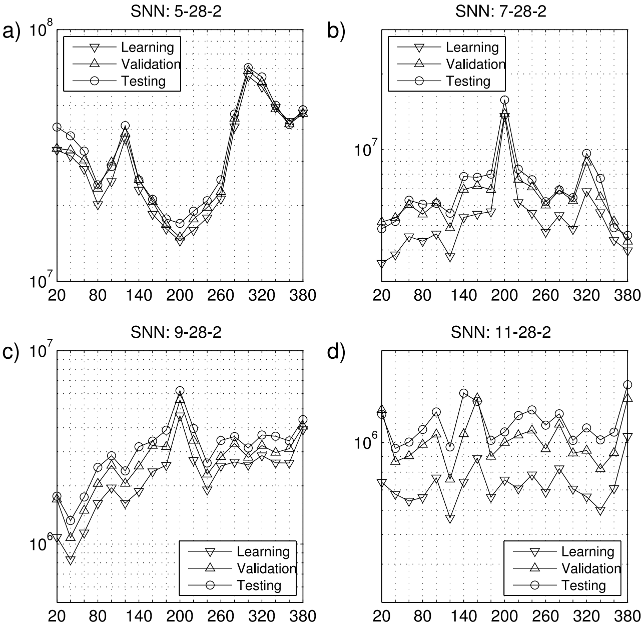


Fig. 3. Optimized locations of control measurement points: a) first, 2) second, c) third and d) fourth point

In case of the beam parameter identification the search of optimized locations was performed by means of a comparatively small network SNN: 5-28-2, assuming $J = 2$ and one control point. The possible location of accelerometers was limited to 19 uniformly distributed points. In Fig. 3a, the piece-wise error function $E(l_{P1})$ is depicted for the training, validation and testing sets. The optimized location $l_{P1}^{opt} = 250\,cm$ was evaluated. The network was then extended to SNN: 7-28-2 in order to fix the evaluated location of the first control point. As can be seen in Fig. 3b that the second locations corresponds to $l_{P2}^{opt} = 475\,cm$. The 3^{rd} and 4^{th} steps needed the networks SNN: 9-28-2 and SNN: 11-28-2 for evaluating the optimized locations $l_{P3}^{opt} = 50\,cm$ and $l_{P2}^{opt} = 150\,cm$, cf. Figs 3c,d.

The optimized location of control points is shown in Fig. 2b after the rearrangement of sequence the control points, corresponding to components of the eigenvectors v_j.

5 Numerical Analysis and Comparison with Existing Results

The computations were carried out for the following input vector:

$$\omega'_{((1+5J)\times 1)} = \{l_K, \Delta f_i, v_{jP} | j = 1, 2, 3; P = 50, 150, 250, 475\,cm\}\,, \qquad (9)$$

which leads to the network architecture SSS: 16-28-2.

The obtained accuracy of neural identification is measured by two errors, i.e. Root Mean Square Errors ($RMSE^i$) and Average Relative Errors (ARE^i), where the superfix denotes outputs $i = l, R$. The errors were computed only for the testing set of patterns:

$$RMSE^i = \sqrt{\frac{1}{T}\sum_{n=1}^{T}(t_n^i - y_n^i)^2}\,, \quad ARE^i = \frac{1}{T}\sum_{n=1}^{T}\left|\frac{t_n^i - y_n^i}{t_n^i}\right| \times 100\%\,. \qquad (10)$$

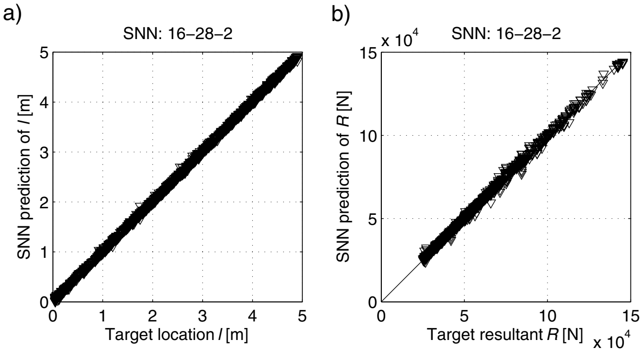


Fig. 4. Distribution of identification points in plane (target, SNN prediction values): a) locations of resultant load l, b) values of resultant load R

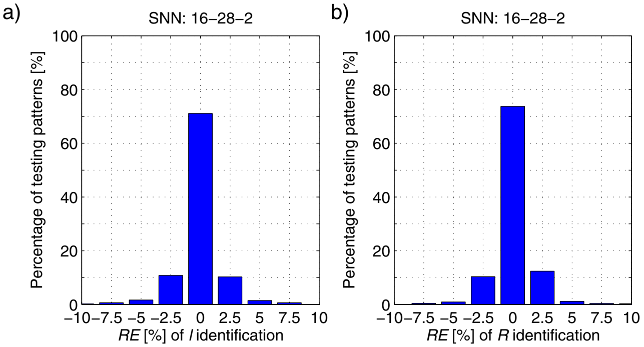


Fig. 5. Histograms of relative errors for identification of load locations l and resultant loads R by the Standard NN

The results of identification of the main load parameters l and R are shown in Fig. 4. The points correspond to considered patterns as the relations $y_n^i\left(t_n^i\right)$. It is evident that the points are placed close to the diagonal $y = t$. These results can also be presented in the form of histograms shown in Fig. 5. The horizontal axis of histograms is related to the relative error $RE_n^i = \left(1 - y_n^i/t_n^i\right) \times 100\%$. The vertical axis shows the percentage of testing patterns to be contained in

Table 1. Testing errors of identification of location l and resultant force R

Architecture of ANNs	Eigenvalues number J	Control measurement points P	Errors for $i = l, R$			
			$RMSE^i$		ARE^i	
			l [m]	$R/1000$ [N]	l [%]	R [%]
SNN: 16-28-2	3	opt. A,B,C,D	0.025	1.02	2.20	1.05
SBNN: 16-23-2	3	n-o. A,B,C,D	0.060	1.90	5.31	2.24
SBNN: 9-23-2	3	n-o. A,B,C	0.101	3.63	10.50	4.48

histogram bins. What is the most important is the central bin for the error interval $[-BW/2, BW/2]$, where BW is an assumed bin width.

It is visible that both the resultant load R and its location l can be identified with errors $|RE^i| \leq 1.25\%$ for about 70% of patterns.

The errors of neural identification are listed in Table 1 not only for the investigated network SNN: 16-28-2 but also for Semi Bayesian NNs SBNN: 16-23-2 and SBNN: 9-23-2, selected from [5]. The SBNNs were analysed for a non-optimized distribution of control points, see Fig. 2a.

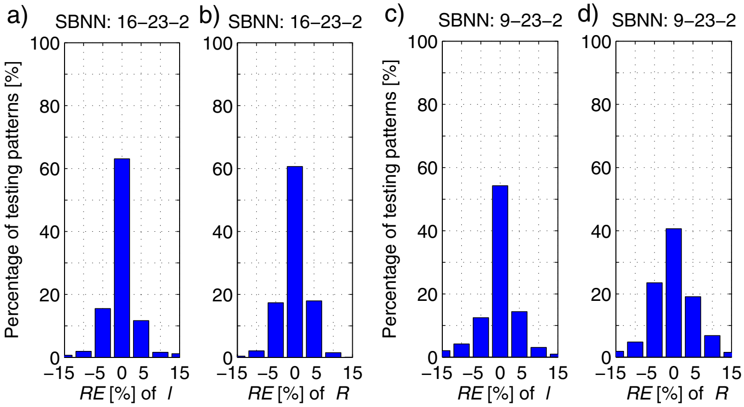


Fig. 6. Histograms of relative errors for identification of load locations l and resultant loads R by Semi Bayesian NNs

It is clear that the optimized location of control measurement points applied in the SNN network gives double decrease of errors identified by the SBNNs. In Fig. 6 the error histograms are depicted for the Semi Bayesian networks SBNN: 16-23-2 and SBNN: 9-23-2. The standard network SNN:16-28-2 gives the value of about 70% correct identified parameters for relative errors $\left|RE_n^i\right| \leq 1.25\%$, see Fig. 5. In case of Semi Bayesian NNs SBNN: 16-23-2, the errors are more than twice as big, since the correct identification of about 60% of patterns was obtained for $\left|RE_n^i\right| \leq 2.5\%$. A smaller network SBNN: 9-23-2 gives for the interval $\left|RE_n^i\right| \leq 2.5\%$ the correct identification for only about 55% patterns with respect to location l and about 40% patterns for prediction of R.

6 Final Remarks

1. In the paper it is shown that the application of optimized locations of control measurement points can significantly improve the accuracy of load parameters identification.
2. The attention was focused on three eigenfrequencies and four control points. The reduction of number of theses parameters should be more deeply recognized.
3. The application of Semi Bayesian and True Bayesian Neural Networks to the analysis of inverse problems is now under development by authors of this paper since they reflect the probabilistic character of the investigated problems. This concerns the parametric identification of load parameters.
4. The paper deals with a theoretical basis of two grants, which are devoted to development of new non-destructive methods of civil engineering structures health monitoring.

Acknowledgments. The financial support of this work by the Polish Grants MNiS N N501 134336 and N N506 4326336 is gratefully acknowledged.

References

1. Miller, B., Ziemianski, L.: The Identification of the Load Causing Partial Yielding on the Basis of the Dynamic Characteristics. Compu. Assist. Mech. Eng. Sci. 13, 627–631 (2006)
2. Uhl, T., Lisowski, W.: Engineering Problems of Structural Modal Analysis. In: ACO, Cracow, Poland (1996) (in Polish)
3. Moran, M.M.: Change of Dynamic Characteristics Due to Plastification. In: Computational Plasticity. Fundamentals and Applications, pp. 1967–1997. Pineridge Press-CIMNE, Swansea-Barcelona (1995)
4. Waszczyszyn, Z., Ziemianski, L.: Neural Networks in the Identification Analysis of Structural Mechanics Problems. In: Mroz, Z., Stavroulakis, G. (eds.) Parameter Identification of Materials and Structures. CISM Lecture Notes, vol. 469, pp. 265–340. Springer, New York (2005)
5. Miller, B.: Application of Semi-Bayesian Neural Networks in the Identification of Load Causing Beam Cross-Section Yielding (Paper submitted for publication in Compu. & Stru.)
6. Dems, K., Mroz, Z.: Identification of Damage in Beam and Plate Structures Using Parameter-Dependent Frequency Changes. Eng. Comp. 18, 96–120 (2001)
7. ADINA R&D Inc., Theory and Modeling Guide. Watertown MS, USA (2009)
8. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Heidelberg (2006)
9. Waszczyszyn, Z., Slonski, M.: Selected Problem of Artificial Neural Networks Development. In: Waszczyszyn, Z. (ed.) Advances of Soft Computing in Engineering. CISM Lecture Notes, vol. 512, pp. 237–316. Springer, New York (2010)
10. Haykin, S.: Neural Networks. A Comprehensive Foundation, 2nd edn. Prentice-Hall, Upper Saddle River (1999)

Architecture of the HeaRT Hybrid Rule Engine

Grzegorz J. Nalepa

Institute of Automatics,
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
gjn@agh.edu.pl

Abstract. In this paper a new rule engine called HeaRT (HeKatE Run Time) is proposed. It uses a custom rule representation called XTT2, which is based on a formalized rule description and allows a formalized analysis of rule quality. The engine is integrated with a complete design environment that provides visual rule design capabilities. The engine supports modularized rule bases and custom inference mechanisms. The rule-based logic can be integrated with the environment using external callback functions in Prolog and Java. In the paper the architecture of the engine is discussed as well as selected implementation issues are given.

1 Introduction

Rules are one of the most successful knowledge representation methods [1]. Therefore, systems build on rules found number of applications if the fields of decision support and diagnosis; for a state-of-the art see [2,3,4,5]. Recently, the Business Rules Approached advocated rule applications in business software [6].

Rule engines have been the main component of classic expert systems shells, e.g. CLIPS [3]. The recent growth of interest in rule-based systems stimulated the development of newer engine implementations, such as Jess[7] and Drools [8]. Every rule engine provides at least a rule language and inference mechanisms. All of the above solutions are based on different variants of the classic Rete algorithm [9]. However, practical implementation of rule-based systems still requires solving *three main problems*: 1) efficient rule representation including modularization of the rule base, 2) rule quality analysis, and 3) rule-based system integration with the environment. For some recent studies see [10,11].

In this paper a new rule engine called HeaRT (HeKatE Run Time) is introduced. It uses a custom rule representation called XTT2 [12], which is based on a formalized rule description [13] and allows a formalized analysis of rule quality. In fact, the engine is integrated with a design environment [14] providing visual rule design capabilities. The engine supports modularized rule bases and custom inference mechanisms. Its architecture addresses the problems identified above.

The rest of the paper is organized as follows: In Sect. 2 the motivation for the HeaRT development is given. The XTT2 rule representation is shortly discussed in Sect. 3. Then, in Sect. 4 the engine architecture and selected implementation details are discussed. The evaluation of the results is given in Sect. 5. The paper ends with concluding remarks as well as directions for future work in Sect. 6.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 598–605, 2010.
© Springer-Verlag Berlin Heidelberg 2010

2 Motivation

Implementation of rule-based systems is a well established field. Number of implementations of rule engines exist, including the classic CLIPS shell, and more recently its Java-based reimplementation – Jess, as well as Drools, tightly integrated with the JBoss application stack. They all share common roots, including a strong assumption of a flat rule base where the Rete-base inference is executed.

In fact, these roots result in their sharing of some old problems. The three main problems in the rule implementation include: 1) efficient rule representation and design, including modularization of the rule base, 2) rule quality analysis, and 3) rule-based system integration with the environment. These three challenges are still areas of active research. In fact, Drools 5 tries to deliver an integrated implementation environment, and all of the mentioned engines provide some kind of rule grouping that might simplify modelling and managing large rule bases. However, these solutions seem to be not sufficient, with the problems of rule quality and effective design mostly not addressed.

The approach that resulted in the implementation of the HeaRT rule engine introduced in this paper is a result of the *Hybrid Knowledge Engineering Project* (HeKatE, see hekate.ia.agh.edu.pl). In this approach the XTT2 rule language, formalized with the use of the ALSV(FD) logic is proposed [13,12]. The language introduces explicit modularization of the rule base, as well as visual rule representation based on the network of decision tables. This allows for an effective visual design on a high level of abstraction, where formal verification of rules is possible. On the other hand it requires a new inference solutions and a custom rule engine, since the classic Rete-like approaches are inappropriate.

In the next section the main aspects of the XTT2 rule representation are given, and then the design and implementation of the HeaRT engine is discussed.

3 Rule Representation and Inference

The main motivation behind the ALSV(FD) attributive logic is the extension of the notational possibilities and expressive power of the tabular rule-based systems [4,13,12]. Some main concepts of the logic are: attribute, atomic formulae, state representation and rule formulation.

After [4] it is assumed that an *attribute* A_i is a function (or partial function) of the form $A_i\colon O \rightarrow 2^{D_i}$. Here O is a set of objects and D_i is the domain of attribute A_i. As we consider dynamic systems, the values of attributes can change over time (or state of the system). We consider both *simple* attributes of the form $A_i\colon T \rightarrow D_i$ (i.e. taking a single value at any instant of time) and *generalized* ones of the form $A_i\colon T \rightarrow 2^{D_i}$ (i.e. taking a set of values at a time); here T denotes the time domain of discourse.

The *atomic formulae* can have the following four forms: $A_i = d$, $A_i = t$, $A_i \in t$, and $A_i \subseteq t$, where $d \in D$ is an atomic value from the domain D of the attribute and $t \subseteq D$, $t = \{d_1, d_2, \ldots, d_k\}$, is a (finite) set of such values. The *semantics* of $A_i = d$ is straightforward – the attribute takes a single value. The semantics of $A_i = t$ is that the attribute takes *all* the values of t (see [12]).

An important extension in ALSV(FD) over previous versions of the logic [4] consists in allowing for explicit specification of one of the relational symbols $=,\neq,\in,\notin,\subseteq,\supseteq,\sim$ and $\not\sim$ with an argument in the table.

From the logical point of view the *state* is represented by the current values of all attributes specified within the contents of the knowledge-base, as a formula:

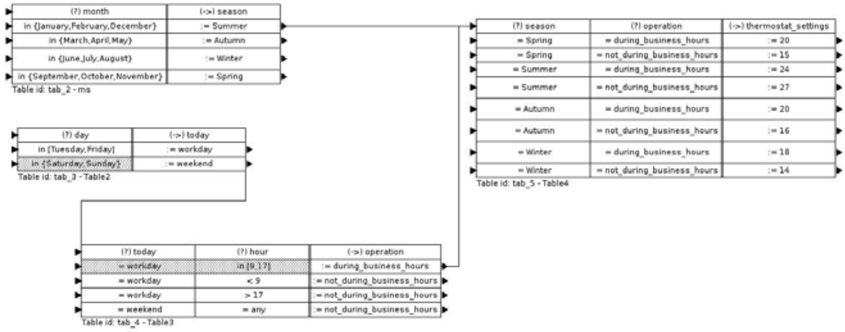
$$(A_1 = S_1) \wedge (A_2 = S_2) \wedge \ldots \wedge (A_n = S_n) \tag{1}$$

where A_i are the attributes and S_i are their current values; note that $S_i = d_i$ ($d_i \in D_i$) for simple attributes and $S_i = V_i$, ($V_i \subseteq D_i$) for generalized ones, where D_i is the domain for attribute A_i, $i = 1, 2, \ldots, n$. An XTT2 rule is:

$$(A_1 \propto_1 V_1) \wedge (A_2 \propto_2 V_2) \wedge \ldots (A_n \propto_n V_n) \longrightarrow RHS \tag{2}$$

where $\propto_i$ is one of the admissible relational symbols, and RHS is the right-hand side of the rule covering conclusions. In practise the conclusions are restricted to assigning new attribute values, thus changing the system state. State changes trigger external callbacks that allow for communication with the environment.

XTT2 (*eXtended Tabular Trees v2*) knowledge representation incorporates attributive table format. Similar rules are grouped within separated tables, and the system is split into a network of such tables representing the inference flow (see Fig. 1) Efficient inference is assured thanks to firing only rules necessary for achieving the goal. It is achieved by selecting the desired output tables and identifying the tables necessary to be fired first. The links representing the partial order assure that when passing from one table to another one, the latter can be fired since the former one prepares an appropriate context knowledge [12].



(?) month	(->) season
in {January,February,December}	:= Summer
in {March,April,May}	:= Autumn
in {June,July,August}	:= Winter
in {September,October,November}	:= Spring

Table id: tab_2 - ms

(?) day	(->) today
in [Tuesday,Friday]	:= workday
in {Saturday,Sunday}	:= weekend

Table id: tab_3 - Table2

(?) today	(?) hour	(->) operation
= workday	in [9,17]	:= during_business_hours
= workday	< 9	:= not_during_business_hours
= workday	> 17	:= not_during_business_hours
= weekend	= any	:= not_during_business_hours

Table id: tab_4 - Table3

(?) season	(?) operation	(->) thermostat_settings
= Spring	= during_business_hours	:= 20
= Spring	= not_during_business_hours	:= 15
= Summer	= during_business_hours	:= 24
= Summer	= not_during_business_hours	:= 27
= Autumn	= during_business_hours	:= 20
= Autumn	= not_during_business_hours	:= 16
= Winter	= during_business_hours	:= 18
= Winter	= not_during_business_hours	:= 14

Table id: tab_5 - Table4

Fig. 1. Example of XTT2 Visual Representation

The visual representation is automatically transformed into HMR (HeKatE Meta Representation), a corresponding textual algebraic notation, suitable for direct execution by the rule engine. An example excerpt of HMR is:

```
xschm th: [today,hour] ==> [operation].
xrule th/1:
  [today eq workday, hour gt 17] ==> [operation set not_bizhours].
xrule th/4:
  [today eq workday, hour  in [9 to 17]] ==> [operation set bizhours].
```

The first line defines an XTT2 table scheme, or header, defining all of the attributes used in the table. Its semantics is as follows: "the XTT table *th* has two conditional attributes: *today* and *hour* and one decision attribute: *operation*". Then two examples of rules are given. The second rule can be read as: "Rule with ID *4* in the XTT table called *th*: if value of the attribute *today* equals (=) value *workday* and the value of the attribute *hour* belongs to the range ($\in$) $< 9, 17 >$ then set the value of the attribute *operation* to the value *bizhours*".

4 HeaRT Architecture and Implementation

HeKatE RunTime (HeaRT) is a dedicated inference engine for the XTT2 rule bases. The architecture of HeaRT can be observed in Fig. 2. The engine is highly modularized. It is composed of the main *inference module* based on ALSV(FD). It supports four types of inference process, Data and Goal Driven, Fixed Order, and Token Driven [12]. The *model management module* allows loading and managing rule models. HeaRT also provides a *verification module*, also known as HalVA (HeKatE Verification and Analysis) [17]. The module implements simple debugging mechanism that allows tracking system trajectory, logical verification of models (several plugins are available, including completeness, determinism and redundancy checks), and syntactic analysis of HMR files using a DCG [18] grammar of HMR. The verification plugins can be run from the interpreter or indirectly from the design environment using the communication module. The *communication and integration module* provides environment integration based on the callbacks mechanism. Another feature is the network-based communication protocol, that allows for both remote access as well as integration with other components of the design environment [14].

The engine is implemented in Prolog, using the SWI-Prolog stack [16]. The main HMR parser is heavily based on the Prolog operator redefinition [18]. A dedicated forward and backward chaining meta interpreter is provided, implementing custom rule inference modes. HeaRT supports Java integration based on callbacks using the SWI-Prolog JPL library. Callbacks can be used to create GUI with JPL and SWING in Java. Another option is to use the SWI XPCE GUI. To make HeaRT integration easier, there are three integration libraries, JHeroic, PHeroic or YHeroic. *JHeroic* library was written in Java. Based on JHeroic one can build applets, desktop application or even JSP services. It is also possible to integrate HeaRT with database using ODBC, or Hibernate. *YHeroic* is a library created in Python. It has the same functionality as JHeroic but is easier to use because of the nature of the Python language. *PHeroic* is the same library but created in PHP5. It can be used in a dynamic web page based on PHP.

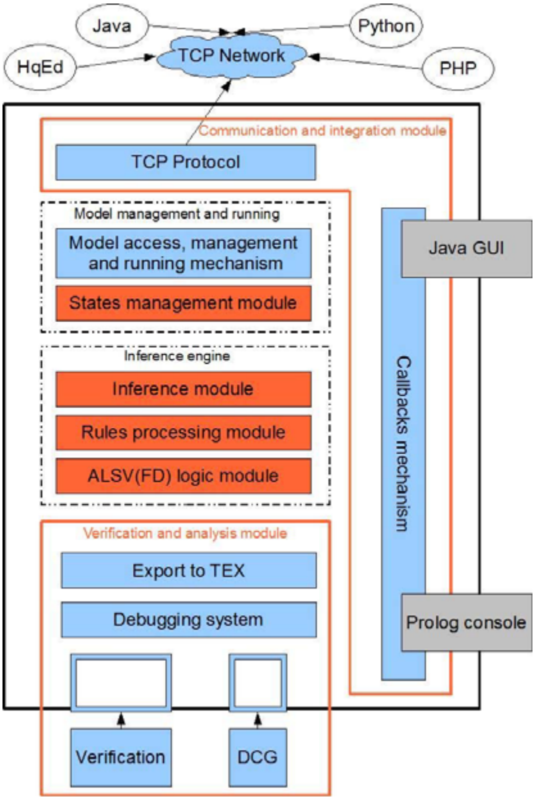


Fig. 2. HeaRT Architecture (see [15,16])

Examples of callback functions are as follows:

```
xcall ask_symbolic_GUI : [AttName] >>>
   (jpl_new('skeleton.RequestInterface',[],T),
   alsv_domain(AttName,Domain,symbolic),
   Params = [AttName|Domain],
   term_to_atom(Params, AtomPar),
   jpl_call(T,request,['callbacks.input.ComboBoxFetcher',AtomPar],Answ),
   atom_to_term(Answ,Answer2,_),
   alsv_new_val(AttName,Answer2)).

xcall tell_console : [AttName] >>>
  (write('Attribute '),
   write(AttName),write(' output value is '),
   xstat(current:[AttName,V]),
   write(V),nl).
```

The first uses the SWI-Prolog JPL library, that dynamically links the Prolog code with Java classes, to display a graphical dialog box. The second is a simple Prolog console-only solution that allows to printout the attribute value.

HeaRT offers a flexible network integration mechanism. It can operate in two modes, stand-alone and as a TCP/IP-based rule logic server. In particular, it allows for integration with a complete rule design and verification environment [14]. It also makes the creation of console or graphical user interface built on the Model-View-Controller design pattern possible. The logic server can also be exposed as a network service in the SOA approach [19].

5 Evaluation

To evaluate the XTT2 method and the HeaRT engine, a number of benchmark rule systems have been modeled in the HeKatE project[1]. They prove the effectiveness of the design approach and rulebase modularization. However, at this point a direct comparison of inference effectiveness with other engines is not obvious due to different inference approach.

When it comes to comparing XTT2 and HeaRT to related approaches the focus is on two important solutions: CLIPS and its Java-based incarnation – Jess, as well as Drools, which inherits some of the important CLIPS features, while providing a number of high-level integration features.

XTT2 provides an expressive, formally defined language to describe rules. It allows for formally described inference, property analysis, and code generation. Additional callbacks in rule decisions provide means to invoke external callbacks in any language. This feature is superior to those of both CLIPS/Jess and Drools. On the other hand, the main limitation of this approach is the state-based system description, where the state is defined as the set of attribute values.

The explicit rule base structure is another feature of XTT. Rules are grouped into decision tables during the design, and the inference control is designed during the conceptual design, and later on refined during the logical design. Therefore, the XTT representation is highly optimized towards rule base modularization. This feature makes the visual design much more transparent and scalable.

In fact all the Rete-based solutions seek some kind of modularization. In the case of CLIPS it is possible to modularize the rule base (see chapter 9 in [3]). It is possible to group rules in modules operating in given contexts, and then provide a context switching logic. Drools 5 offers Drools Flow that allows to define rule set and simple control structure determining their execution. In fact this is similar to the XTT-based solution. However, it is a weaker mechanism that does not correspond to table-based solution.

A complete design process seems to be in practice the most important issue. Both CLIPS and Jess are classic expert system shells, providing rule languages, and runtimes. They are not directly connected to any design methodology. The rule language does not have any visual representation, so no complete visual editors are available. Implementation for these systems can be supported by a number of external environments (e.g. Eclipse). However, it is worth emphasizing, that these tools do not visualize the knowledge contained in the rule base.

[1] See `https://ai.ia.agh.edu.pl/wiki/hekate:cases:start`

It is crucial to emphasize, that there is a fundamental difference between a graphical user interface like the one provided by generic Eclipse-based solutions, and *visual design support and specification* provided by languages such as XTT for rules, and in software engineering by UML.

6 Concluding Remarks and Future Work

The main motivation for the research presented in this paper is to overcome some persistent problems of rule design and implementation including efficient rule representation and design, rule quality analysis, and rule-based system integration with the environment. The paper discusses the implementation of the HeaRT rule engine, a result of the *Hybrid Knowledge Engineering Project* (HeKatE, see hekate.ia.agh.edu.pl). In this approach the XTT2 rule language, formalized with the use of the ALSV(FD) logic is proposed [13,12]. It introduces explicit modularization of the rule base, as well as visual rule representation based on the network of decision tables. This allows for an effective visual design on a high level of abstraction, where formal verification of rules is possible. The engine supports new inference algorithms for modularized rue bases, as well as flexible environment integration features, using callback and network communication.

Future work includes a tighter design tool integration, as well as modeling complex cases in order to identify possible limitations of the XTT2 methodology. Providing a comparative studies modelling the same cases in XTT, CLIPS and Drools is planned. This would also allow for a more accurate comparison of the engine efficiency. Another direction is building a complete library of callback functions for common user interface elements in different languages. This would simplify interface integration on different platforms.

Acknowledgements. The Author wishes to thank his Master students Szymon Bobek and Michł Gawędzki who implemented important components of the designed engine as a part of their Master Thesis [15].

References

1. van Harmelen, F., Lifschitz, V., Porter, B. (eds.): Handbook of Knowledge Representation. Elsevier Science, Amsterdam (2007)
2. Liebowitz, J. (ed.): The Handbook of Applied Expert Systems. CRC Press, Boca Raton (1998)
3. Giarratano, J., Riley, G.: Expert Systems. Principles and Programming, 4th edn. Thomson Course Technology, Boston (2005), ISBN 0-534-38447-1
4. Ligęza, A.: Logical Foundations for Rule-Based Systems. Springer, Heidelberg (2006)
5. Giurca, A., Gasevic, D., Taveter, K. (eds.): Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches. Information Science Reference, Hershey (2009)
6. Ross, R.G.: Principles of the Business Rule Approach, 1st edn. Addison-Wesley Professional, Reading (2003)

7. Friedman-Hill, E.: Jess in Action, Rule Based Systems in Java. Manning (2003)
8. Browne, P.: JBoss Drools Business Rules. Packt Publishing (2009)
9. Forgy, C.: Rete: A fast algorithm for the many patterns/many objects match problem. Artif. Intell. 19(1), 17–37 (1982)
10. Nalepa, G.J.: Languages and tools for rule modeling. In: Giurca, A., Dragan Gasevic, K.T. (eds.) Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches, pp. 596–624. IGI Global, Hershey (2009)
11. Ligęza, A., Nalepa, G.J.: Rules verification and validation. In: Giurca, A., Dragan Gasevic, K.T. (eds.) Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches, pp. 273–301. IGI Global, Hershey (2009)
12. Nalepa, G.J., Ligęza, A.: HeKatE methodology, hybrid engineering of intelligent systems. International Journal of Applied Mathematics and Computer Science (2010) (accepted for publication)
13. Nalepa, G.J., Ligęza, A.: XTT+ rule design using the alsv(fd). In: Giurca, A., Analyti, A., Wagner, G. (eds.) ECAI 2008: 18th European Conference on Artificial Intelligence: 2nd East European Workshop on Rule-based applications, RuleApps2008, July 22, pp. 11–15. University of Patras, Patras (2008)
14. Nalepa, G.J., Ligęza, A., Kaczor, K., Furmańska, W.T.: HeKatE rule runtime and design framework. In: Adrian Giurca, G.W., Nalepa, G.J. (eds.) Proceedings of the 3rd East European Workshop on Rule-Based Applications (RuleApps 2009) Cottbus, Germany, September 21, pp. 21–30 (2009)
15. Bobek, S., Gawędzki, M.: Design and implementation of a runtime environment for the XTT^2 rule representation method. Master's thesis, AGH University of Science and Technology (July 2009); Supervisor: G. J. Nalepa
16. Nalepa, G.J., Bobek, S., Gawędzki, M., Ligęza, A.: HeaRT Hybrid XTT^2 rule engine design and implementation. Technical Report CSLTR 4/2009, AGH University of Science and Technology (2009)
17. Ligęza, A., Nalepa, G.J.: Proposal of a formal verification framework for the XTT^2 rule bases. In: Tadeusiewicz, R., Ligęza, A., Mitkowski, W., Szymkat, M. (eds.) CMS 2009: Computer Methods and Systems: 7th Conference, Kraków, Poland, November 26–27, pp. 105–110. AGH University of Science and Technology, Cracow, Oprogramowanie Naukowo-Techniczne (2009)
18. Bratko, I.: Prolog Programming for Artificial Intelligence, 3rd edn. Addison-Wesley, Reading (2000)
19. Erl, T.: Service-Oriented Architecture (SOA): Concepts, Technology and Design. Prentice Hall PTR, Englewood Cliffs (2005)

Using Extended Cardinal Direction Calculus in Natural Language Based Systems

Jedrzej Osinski

Faculty of Mathematics and Computer Science
Adam Mickiewicz University, Poznan, Poland
josinski@amu.edu.pl

Abstract. The Cardinal Direction Calculus (CDC) is one of the most popular qualitative technique for spatial reasoning. The extended CDC (XCDC) can be successfully used for representation and reasoning about the spatio-temporal aspects of complex events. This paper discusses the problem of composing spatial relations in this formalism within systems using natural language input. We present the usage of the classic composition algorithm for both external and internal direction relations that can be driven from sentences in a natural language. Also some other solutions are introduced to improve the precision of the information processed by a system.

Keywords: natural language processing, human-computer interaction, cardinal direction calculus, spatio-temporal reasoning.

1 Introduction

The Cardinal Direction Calculus (CDC) is a widely known technique of qualitative calculus for spatial reasoning. This directional formalism, which makes abstraction from quantitative knowledge, can be successfully used for describing spatial relations between objects, computing positions [3] and for assessing similarity between spatial scenes [6]. However a natural language input generates the series of problems. In particular, the natural language is generally characterized by low precision. Moreover it is necessary to reach the compromise in a computer representation of the real-world relations: it is important to avoid losing any potential information (so we should assume e.g. all the possible localization of an object), but on the other hand too general fact has no value (e.g. the information that an object can be everywhere). We will discuss this problem more carefully and present some suggested solutions in the next sections.

A spatial relation between two events is represented by a pair of direction-relation matrices from the Cardinal Direction Constraints technique originally proposed in [5]. The key idea of that formalism is based on dividing the plane around the reference object (i.e. the object from which the direction relation is determined) into nine regions named after the geographical directions: *NW*, *N*, *NE*, *W*, *O* (central region meaning the same location), *E*, *SW*, *S* and *SE*. This areas, called direction tiles, are closed, unbounded (except for *O*), their

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 606–613, 2010.
© Springer-Verlag Berlin Heidelberg 2010

interiors are pairwise disjoint and their union is the whole plane. Directions between the reference object A and target object B are represented in a 3 x 3 matrix denoted by dir(A,B). It is worth noticing that this classical model is often extended in different aspects, e.g. in [3] we can find reference to the four bordering lines on A at the top, bottom and both sides. The extension increases precision of information (and could be easily applied as an extension of the proposed representation), however it would not be useful for systems whose knowledge is based on human senses and natural language (two domains which are naturally characterised by low precision). Moreover, not every context needs such a degree of details e.g. if we describe interactions between people, in most cases it is not really important whether they have touched each other (e.g. while shaking hands) or just were standing next to each other - the significant fact is they have actually met. It is widely known that information which is too precise can make deduction difficult.

We define the direction-relation matrix as follows:

$$dir(A,B) = \begin{bmatrix} f(NW(A) \cap B) & f(N(A) \cap B) & f(NE(A) \cap B) \\ f(W(A) \cap B) & f(O(A) \cap B) & f(E(A) \cap B) \\ f(SW(A) \cap B) & f(S(A) \cap B) & f(SE(A) \cap B) \end{bmatrix},$$

$$where f(x) = \begin{cases} 0, if\ Interior(X) \neq \emptyset \\ 1, if\ Interior(X) = \emptyset \end{cases}.$$

Now we can define the spatial relation between two events A and B as follows:

$$SR(A,B) = [dir(A,B), dir(B,A)].$$

The above definition describes a formalism called Extended CDC (XCDC, originally introduced in [10]) enclosed (for simplicity) to the spatial aspects of the mentioned problems.

As can be seen we concentrate our analysis on binary matrices. However it is also possible to use matrices in which each element is a real number in the interval [0,1] and all the entries sum up to 1. This would lead to the fuzzy CDC. Again we come across the question of precision. In this paper we decided to use the Boolean matrix taking into consideration the low precision of natural language and dynamic environments. In fact no human stands exactly in one place all the time (even a talking person usually shifts one's weight from foot to foot or walks in small circles).

2 Defining Basic Direction Relations

The composition of relations is an important feature of a system for reasoning about the spatio-temporal aspects of complex events. Such a functionality can be used e.g. to calculate possible location of an object. If a system is dedicated to support a staff responsible for public security no mistakes are acceptable. What is more if we ask for the localization of a dangerous person it is better for us to get few more possible places to check rather than have one of them missing.

That is why it is very important to return to a user all the answers that suit his/her query. However while using the classic composition algorithm originally described in [12] some results can be lost if direction relations are not properly defined. Before we will give an example it is necessary to recall two definitions presented in the mentioned paper. We will also use the same notation (N for [0,1,0,0,0,0,0,0,0], O for [0,0,0,0,1,0,0,0,0], *NW:N:W* for [1,1,0,1,0,0,0,0,0], etc.) to simplify the calculation done below.

Let R be a basic cardinal direction relation. Then:

Def 1. The bounding relation of R, denoted by Br(R), is the rectangular relation with the smallest number of tiles that includes R.

Def 2. Most(N, R) is a rectangular relation formed by the northernmost tiles of a relation R.

The notation $\delta(R_1,\ldots,R_k)$ is a shortcut for disjunctive relations that can be constructed by combining single-tile relations $R_1,\ldots,R_k$. U_{dir} is the universal cardinal direction relation.

For R_1 being a single-tile relation and R_2 - a cardinal direction relation the classic composition algorithm can be defined by the equation:

$$R_1 \circ R_2 = (R_1 \circ Most(R_1, Br(R_2)).$$

Now let us consider the following example. Suppose there is an information about two relation between objects X, Y, Z sent to a system as two sentences in a natural language:

(S1) *X is to the west of Y.*
(S2) *Z is to the east of X.*

Now according to the basic CDC formalism we define the relation *to the west of* as a direction-relation matrix W and similarly, the relation *to the east of* as E. Now let us calculate the relation between Z and Y, denoted R (which was not directly entered to the system). This is necessary if a user wants to know the location of the object Z with the reference to the object Y (e.g. *I am in Y. Where can I find Z?*). According to the classic algorithm:

$$R = W \circ E = W \circ Most(W, Br(E)) = W \circ Most(W, E) = W \circ E = \delta(W, O, E).$$

We can notice that the results do not cover all the possible relations between the objects. Fig. 1 shows an example of such a situation. The sentence in a natural language does not contain information about the size of the target object. That is why it is necessary to assume all the possible cases. The other aspect of this problem is a very specific way of how people talk about spatial relations. It was carefully discussed in [11] where results of a language experiment were introduced. Participants were asked to describe in a natural language the situations presented on simple pictures corresponding to the direction relations. In particular, when a relation *NW:N:NE* had been shown, 84.5% used the sentence *'to the north of'* and only 9.9% said it was *'to the north-west, north and north-east of'*.

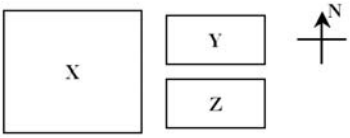


Fig. 1. The example of spatial relations between objects

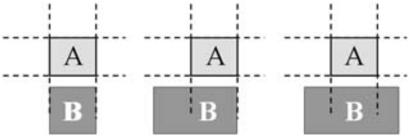


Fig. 2. Different possible situations for *dir(B,A)=N*

On the other hand a basic direction relation does not provide any information about the size of a reference object. Fig. 2 shows different possible arrangements of objects A and B in the plane such that all of them can be defined by the same direction-relation matrix *dir(B,A)=N*. These conclusions leaded us to define the basic language cardinal direction relation (LR) as shown in Tab. 1. Now let us calculate the relation between Z and Y again (see the example above), this time using definitions proposed in Tab. 1 and XCDC formalism ($SR = [R_1,\ R_2]$). According to the classic algorithm:

$$\begin{aligned}
&R_1 = NE:E:SE \circ NW:W:SW = \\
&(NE \circ Most(NE, Br(NW:W:SW))) \cup (E \circ Most(E, Br(NW:W:SW))) \\
&\cup(SE \circ Most(SE, Br(NW:W:SW))) = (NE \circ Most(NE, NW:W:SW)) \\
&\cup(E \circ Most(E, NW:W:SW)) \cup (SE \circ Most(SE, NW:W:SW)) = \\
&= (NE \circ NW) \cup (E \circ NW:W:SW) \cup (SE \circ SW) = \\
&= \delta(NW, N, NE) \cup U_{dir} \cup \delta(SE, S, SW) = U_{dir}.
\end{aligned}$$

Similarly $R_2 = R_1$. As can be noticed $N \in R_1$ what means that the situation shown in the Fig. 1 would not be omitted by a system.

3 Defining Internal Direction Relations

Apart from the conventional cardinal direction relations (which were discussed above) there exist Internal Cardinal Direction (ICD) relations which are used to describe the situation when one object/region contains another object (or objects). This formalism was carefully analysed in [8] where also three ICD models called ICD-5, ICD-9 and ICD-13 were introduced. These models are characterized by a different degree of details. To choose the one that fit best our particular application it is necessary to analyse two aspects: the scale of the container objects and the spatial distribution characteristics of target objects. We will now present how the XCDC formalism can be simple used for represent ICD relations. Suppose there is a sentence in a natural language *B is in the north-west of A*. As can be noticed this statement describes the topological situation when the object B is localized somewhere in the north-west part of the region A (Fig. 3a). Let us use the classic CDC definition twice: first for B being a target object (Fig. 3b), then for B treated as a reference object (Fig. 3c). Similarly we can define all the basic language ICD relation (ILR) as shown in

Table 1. The basic language relations (*LR*) and internal language relations (*ILR*) between objects B and A

	LR		*ILR*	
B *LR/ILR* A	$dir(A, B)$	$dir(B, A)$	$dir(A, B)$	$dir(B, A)$
to/in the north of	NW:N:NE	SW:S:SE	O	O:S
to/in the south of	SW:S:SE	NW:N:NE	O	N:O
to/in the west of	NW:W:SW	NE:E:SE	O	O:E
to/in the east of	NE:E:SE	NW:W:SW	O	W:O
to/in the north-west of	NW:N:W	SE	O	O:E:S:SE
to/in the north-east of	N:NE:E	SW	O	W:O:SW:S
to/in the south-west of	W:SW:S	NE	O	N:NE:O:E
to/in the south-east of	E:S:SE	NW	O	NW:N:W:O

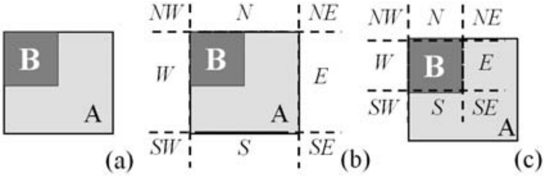


Fig. 3. 'B is in the north-west of A'

the Tab. 1. Both presented tables together give a full model that can be used in systems with natural language input no matter whether an informer describes a real-world situation by internal or external direction relations. What is more, if necessary, further relations can be easily defined. For example we can define a language relation *inside* (as in the sentence *N is inside of M*) as follows: $dir(M, N)$, $dir(N, M) = U_{dir}$. The reason of such a definition is the lack of information about the exact location of N within the region M. Thus we need to assume all the possible places which can be interpreted as a sum of all the basic internal directions. It is worth to mention that even more relations can be generated if we analyse also the aspect of time i.e. the full XCDC.

Now we will show how the classic composition algorithm can be simply implemented for composing relations of different types.

Let us consider the following example. Suppose there is an information about two relation between objects X, Y, Z sent to a system as two sentences in a natural language:

(S3) *X is in the west of Y.*
(S4) *Y is to the south of Z.*

Now let us calculate the relation between X and Z, denoted $SR = [R_1, R_2]$, according to the classic algorithm:

$$R_1 = O \circ SW : S : SE = O \circ Most(O, Br(SW : S : SE)) =$$
$$= O \circ Most(O, SW : S : SE) = O \circ SW : S : SE = SW : S : SE.$$
$$R_2 = O : E \circ NW : N : NE =$$
$$(O \circ Most(O, Br(NW : N : NE))) \cup (E \circ Most(E, Br(NW : W : NE))) =$$
$$= (O \circ Most(O, NW : N : NE)) \cup (E \circ Most(E, NW : W : NE)) =$$
$$= (O \circ NW : N : NE) \cup (E \circ NE) = NW : N : NE \cup NE = NW : N : NE.$$

The results of the calculations are exactly what we could expected if we imagine the spatial situation described by a user (see Fig. 4). No information was lost. What is more in this particular example relation *SR* can be directly translated into the language relation *LR = 'to the south of'* so the reply of a system can be presented as a following sentence: *X is to the south of Z*. However the composition of relations not always is can be directly interpreted (see the previous example). Thus the generation of text replies (which fulfill the requirements of a specific user) is a totally different aspect which will not be discussed in this paper.

4 Enclosing Results of a Composition

As was already discussed the natural language is characterized by low precision. That was the main reason why we decided to propose the definitions of language relations presented in two previous sections. The direction-relation matrices which correspond to these relations are less precise than single tiles. Such a solution protect us from losing some potentially important spatial information. On the other hand information (or result of a composition) which is too general is useless. That is why we will now present the solutions that can enclose the results of a composition.

The first suggestion is connected with disjunctive relations which can be generated by the classic algorithm. Let us consider $\delta(NW,\ W,\ SW)$ as an example. This notation is a shortcut for $\{NW,\ W,\ SW,\ NW{:}W,\ W{:}SW,\ NW{:}W{:}SW\}$. Analysing all of these relations seems necessary however it is worth to check what was the goal of a calculation. If a user asked a system for possible localization of a danger person it is enough to find the sum of these relations. In this example *NW:W:SW* is an exhaustive information. No information was lost, e.g. a danger person cannot be located in any other place. All the other relations cover the same spatial area. In general in similar applications we can use the following simplification:

$$\delta(R_1, \ldots, R_k) = R_1 : \ldots : R_k. \tag{1}$$

Now let us discuss the aspect of calculating the relation between two objects which was not directly entered to a system. Of course it is necessary to use the composition algorithm. However choosing the set of relations which are to be composed is not so obvious. Especially in a system describing complicated environment there are usually many different possible sequences of relations which,

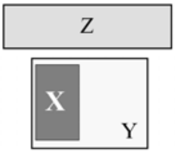


Fig. 4. The spatial situation described by the sentences (S3) and (S4)

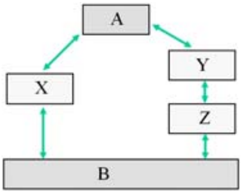


Fig. 5. Two sequences of compositions

if composed correctly, can lead us to a result. It is natural that the shorter a sequence is the more precise is a result (each single composition increase the number of tiles that must be taken into consideration). However to enclose a result we suggest to run the composition algorithm for few different sequences of relations (at least two) and then calculate a logical conjunction of the corresponding tiles of the direction-relation matrices.

Let us consider the following example of this solution. For simplicity we will analyse only a single (one) direction-relation matrix $dir(A, B)$. The task is to find the relation between objects A and B which are not directly set. The relations between the rest of the objects are graphically presented on Fig. 5. We will use the algorithm twice, then the conjunction of the achieved matrices will be calculated:

$$\begin{aligned} dir_1(B, A) &= dir(X, A) \circ dir(B, X) = \\ &= NE \circ N = \delta(NE, N) =^{(1)} NE : N. \\ dir_2(B, A) &= dir(Y, A) \circ (dir(Z, Y) \circ dir(B, Z)) = \\ &= NW \circ (N \circ N) = NW \circ N = \delta(NW, N) =^{(1)} NW : N. \\ dir(B, A) &= dir_1(B, A) \cap dir_2(B, A) = NE : N \cap NW : N = N. \end{aligned}$$

As can be noticed the conjunction of the compositions is much more precise than its elements. In general the more compositions are calculated the more precise a result is.

5 Conclusion

We have presented the practical usage of the XCDC formalism in a system with the natural language input. We have also introduced the XCDC definitions for the external and internal qualitative language relations, so the presented solution combines the features of both CDC and ICD techniques. Consequently the proposed model can be implemented in a system no matter which kind of relations a potential informer uses to describe a real-world situation. We have also analysed the examples of the usage of the classic composition algorithm to present its application in the full XCDC. Finally we have described some techniques for improving the precision of results.

Acknowledgements

This research was partially covered by the Polish Government grant R00 028 02 "Text processing technologies for Polish in application for public security purposes" (2006-2010) headed by Z. Vetulani, within the Polish Platform for Homeland Security.

References

1. Allen, J.F.: Maintaining Knowledge about Temporal Intervals. Artificial Intelligence and Language Processing 26(11), 832–843 (1983)
2. Balbiani, P., Condotta, J., Farinas del Cerro, L.: A new tractable subclasses of the rectangle algebra. In: Proceedings of 16th International Joint Conference on Artificial Intelligence (1999)
3. Cicerone, S., Di Felice, P.: Cardinal directions between spatial objects: the pairwise-consistency problem. Information Sciences 164(1-4), 165–188 (2004)
4. Frank, A.: Qualitative Spatial Reasoning about Cardinal Directions. In: Proceedings of Tenth International Symposium on Computer-Assisted Cartography(Auto-Carto 10) (1991)
5. Goyal, R.K., Egenhofer, M.J.: Cardinal directions between extended spatial objects. IEEE Transactions on Knowledge and Data Engineering (2001)
6. Goyal, R.K., Egenhofer, M.J.: Similarity of Cardinal Directions. In: Jensen, C.S., Schneider, M., Seeger, B., Tsotras, V.J. (eds.) SSTD 2001. LNCS, vol. 2121, pp. 36–55. Springer, Heidelberg (2001)
7. Ligozat, G.: Reasoning about Cardinal Directions. J. Visual Languages and Computing 9, 23–44 (1998)
8. Liu, Y., Wang, X., Jin, X., Wu, L.: On Internal Cardinal Direction Relations. In: Cohn, A.G., Mark, D.M. (eds.) COSIT 2005. LNCS, vol. 3693, pp. 283–299. Springer, Heidelberg (2005)
9. Navarette, I., Morales, A., Sciavicco, G.: Consistency Checking of Basic Cardinal Constraints over Connected Regions. In: IJCAI 2007 (2007)
10. Osinski, J.: Extending the Cardinal Direction Calculus to a Temporal Dimension. In: Proceedings of FLAIRS-22 Conference, Sanibel Island, USA (2009)
11. Osinski, J.: The experimental analysis of the natural language used for describing qualitative spatial relations between objects. In: Proceedings of LTC 2009 Conference, Poznan, Poland (2009)
12. Skiadopoulos, S., Koubarakis, M.: Composing cardinal direction relations. In: Jensen, C.S., Schneider, M., Seeger, B., Tsotras, V.J. (eds.) SSTD 2001. LNCS, vol. 2121, pp. 299–317. Springer, Heidelberg (2001)
13. Zhang, X., Liu, W., Li, S., Ying, M.: Reasoning with Cardinal Directions: An Efficient Algorithm. In: Proceedings of AAAI 2008 Conference (2008)

Metamodelling Approach towards a Disaster Management Decision Support System

Siti Hajar Othman and Ghassan Beydoun

School of Information Systems and Technology, Faculty of Informatics,
University of Wollongong, Wollongong NSW 2522, Australia
{sho492,beydoun}@uow.edu.au

Abstract. Expertise in disaster management (DM) is scarce and often unavailable in a timely manner. Moreover, it is not timely shared as it is often perceived as too tied to kinds of events (floods, bushfires, tsunamis, pandemic or earthquake), leading to catastrophic consequences. In this paper, we lay out a framework to create a decision support system to unify, facilitate and expedite access to DM expertise. We observe that many DM activities are actually common even when the events vary. We provide ontology as a metamodel to describe the various DM activities and desired outcomes. This ontology will serve as a representational layer of DM expertise leading to a DM decision support system based on combining and matching different DM activities according to the disaster on hand.

Keywords: Metamodelling, Ontology, Disaster management, Disaster models, Common concepts, Decision support system.

1 Introduction

The increasing number of disasters recently, such as earthquakes, tsunamis, floods, bushfires, air crashes, epidemic, have posed a huge challenge not only to population at large, but also to public services and agencies tasked with activities relating to preventing and managing disaster responses. Recent failures can be easily identified in the management of the Swine-Flu (H1N1) pandemic hitting Australian shores in large numbers through cruise ships or in the devastating communication failures in the recent bushfires in Victoria (Australia). Many such failures are due to expertise not being available in a timely manner. This is partly due to inability to recognize and identify correct expertise, as it is often perceived as too tied to kinds of events (floods, bushfires, tsunamis or earthquake). Potential of reusing expertise is often overlooked leading to catastrophic consequences. In this paper, we present an approach to unify DM knowledge to create a DM Decision Support System (DSS) that combines and matches different DM activities to suit the disaster on hand.

Failures in preventing disasters or failures in their subsequent management are rarely caused by a single factor. They are often due to an accumulation of complex chain of events and often accompanied by changes in external environment factors. Hence, it is common wisdom that no two disasters are exactly the

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 614–621, 2010.
© Springer-Verlag Berlin Heidelberg 2010

same and that every disaster requires its own management process. On the other hand, the way disasters impact human lives and business processes may well be similar and responses are often transferrable between disasters. Evacuation of personnel for example is a DM action that is applicable in many disaster situations. This paper aims to use a generic representational layer (a metamodel) to give a unified view of common concepts that apply in various disasters. We use existing DM models [1-5] as a starting point towards creating a repository of past DM experiences to be stored as reusable components and expressed using concepts identified in a generic DM metamodel. This will be the first to create a DSS to enable formulating DM approaches as new situations arise.

The approach proposed in this paper is inspired by a software engineering knowledge management practice known as method engineering which involves storing various software methodologies as a collection of reusable process fragments for later reuse to create hybrid methodologies as new software development projects arise. In DM, the first step and the focus of this paper is to appropriately represent DM knowledge and to warehouse DM knowledge in an appropriate form to later allow mixing and matching DM experiences. The appropriate representation of DM knowledge will enable the creation of a repository of DM experiences. Interfacing this to a Decision Support System that takes as input new disaster parameters, will assist in deciding the best DM approach by combining various actions from previous DM experiences.

2 Related Work and Decision Support System Architecture

Our work draws on research from method engineering, experience factories in software engineering, metamodelling and DM literature produced by World Health Organisation and Emergency Management Australia. *Method engineering* is itself an application of knowledge based technology typically underpinned by software engineering results for completion of knowledge representation and acquisition. Our work develops existing tentative attempts to represent DM knowledge in a reusable form to give a unified point of access supported by an intelligent DSS. In particular, later in this paper, we illustrate our unification approach by presenting an initial metamodel that we believe could generalize most of the concepts used in existing DM models. We first detail our approach further and how it relates and draws on existing research. DM as our core application domain is defined as a management of all aspects of planning and responding to all phases in disaster. These phases include mitigation, preparedness, response and recovery activities [6]. This definition includes the management of risks and consequences of disaster. Large disasters cut across many boundaries including organizational, political, geographical and sociological. This presents serious challenges in interoperability between various teams and creates difficulties in collaboration and cooperation across authorities, countries and systems. Moreover, data collection and integration problems arise as various technologies and tools are typically involved in data gathering and monitoring e.g. Global Positioning Systems (GPS), Geographical Information Systems (GIS), data collection platforms and early warning systems. A solid, general and global of coordination on how people work and data is exchanged before, during and after disaster through is still inadequate. We propose the use of metamodelling to uncover and make explicit the key aspects of activities, cooperation and components in DM. Metamodelling is a central activity promoted by the efforts of the Object Management Group (OMG) in the software development paradigm of the Model Driven Architecture (MDA). It aims to create interoperable, reusable, portable software components and data models based. In this software development context, a metamodel is a fundamental building block that makes statements about the possible structure of models. It is usually defined as a set of constructs of a modelling language and their relationships, as well as constraints and modelling rules without necessarily the concrete syntax of the language [2].

Surveying a number of existing DM metamodels, we observed that some concepts represent a similar DM activities or actions but expressed in different terms as follows:

- **Circular Model for Disaster [7]**: Disaster mitigation, Disaster prevention, Disaster preparedness, Warning,Disaster, Emergency Response, Rehabilitation, Reconstruction, Development;
- **A Comprehensive Conceptual Model For Disaster Management**: Hazard assessment, Strategic planning, Risk management, Mitigation, Preparedness, Response, Recovery, Monitoring and evaluation;
- **Ibrahim-Razi Model**: Inception of errors, Accumulation of errors, Warnings, Disaster impending stage, Triggering event Emergency state, Disaster, Normal state;
- **Traditional DM Cycle Model**: Mitigation, Preparedness, Disaster Impact, Reconstruction, Rehabilitation;

For example, in a Circular Model for Disaster [7], the terminology '*Emergency Response*' is being used to represent the response and rescue activity of disaster victims. But, the same activity however is represented by using '*Emergency State*' in Ibrahim-Razi Model. Managing knowledge of this complex domain is hard. As noted in [4], cyber infrastructure is proposed to manage the large number of activities involved in DM. Benefits of a unified metamodel include: domain concepts can be easily presented to newcomers, manage to increase a portability of models across supportive modelling tools, could create better communication amongst practitioners and research could then focus on improving and/or realizing a unified body of knowledge [2].

Developing a DM metamodel is our first step towards creating a DSS to unify, facilitate and expedite access to DM expertise. This metamodel will describe the various DM activities and desired outcomes and serve as a representational layer of DM expertise, enabling an appropriate DM DSS based to guide combining and matching different DM activities according to the disaster scenario on hand as defined by the disaster itself, the involved stakeholders and various rescue teams available. The DM metamodel will be complemented with a *Disaster Retrieval Model* that will be used to choose appropriate procedures and suit with different kinds of disaster (natural or man-made) on hand. Figure 1 illustrates our integration and a DSS platform which will be context independent. For instance, different countries have their own organization in coordinating and act as an advisory board for handling disaster activities. For example, in Australia, we have EMA (*Emergency Management Australia*), United States of America has FEMA (*Federal Emergency Management Agency*) and Canada has the PSC (*Public Safety Canada*). Hence for the purpose of developing our DM metamodel, models of different DM activities as applied by different countries are to be combined and stored into one database namely *DM Activities Repository.* This will be a collection of organizational, operational, planning, logistics and administration procedures and policies executed by these countries through their DM processes. These will be identified and organized according to the DM metamodel consisting of common concepts used in all four disaster phases.

The generic DM metamodel based on identified common concepts will become a *destination point* of scattered concepts used in many DM activities worldwide. As we will later discuss, a process towards concept generalization will be applied to make our DM metamodel more applicable. Activities from different sources (and countries) will be stored as *Procedure Fragments* in the DM knowledge repository. The DSS will assist in deriving the best disaster procedure fragment solution according to the disaster on hand. It will use a set of rules that will specifically determine what is the best solution based on disaster description input entered by a Disaster Manager, as a main user of the system and the repository. We adapt model-based reasoning techniques in the way we determine the best decision solution for our DSS system. However the details discussion on this technique is out of the scope of this paper. DSS with the Disaster Retrieval Model and the DM Metamodel will form the *DM Activities Integration System.* As an example, assume 'DM P1', 'DM P2', 'DM P3' and 'DM P4' are procedures of *Evacuation*, *Mitigation Analysis*, *Rescue* and *Recovery* respectively. These four procedure fragments will commonly occur in real DM scenarios, and will have various instantiations in the repository of DM activities. In various scenarios, the DSS will produce solutions that combine various instances of those procedures differently. For example, it may that **Best Bushfire Decision Solution** is a combination of *An Evacuation Procedure from France*,*A Mitigation Analysis from Australia*, *A Rescue Procedure from US* and *A Recovery Procedure from France.* To produce the DSS system and a populated DM knowledge repository, the first step is to construct the DM metamodel. This will be using existing DM metamodels (to be described in the Section 3) and emergency and DM literature (described early in this section). Using the metamodel, DM procedures will be classified and formulated into a unified repository. A knowledge based interfaced to the repository will be finally be developed to support retrieval and integration of the procedure fragments. This paper undertakes significant work towards the creation of a DM metamodel which will be the focus of the rest of this paper. In the next section, we overview existing relevant metamodels and describe the process of formulating our DM metamodel, before the metamodel is presented in Section 4.

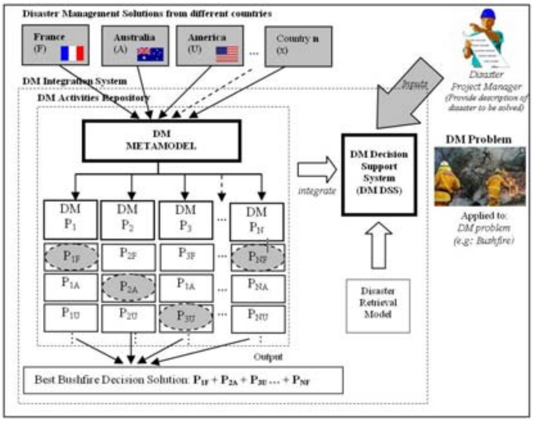


Fig. 1. A framework model of motivation towards Disaster Management DSS

3 DM Metamodelling Process and Existing DM Models

To create our DM metamodel, we adapt a metamodelling approach from the work used to develop a Framework for Agent Modelling Language (FAML) in [2] and [3]. The approach consists of these steps:

- **Step 1**: Extraction of general concepts relevant to any DM model using relevant literature [1-5] which will be reviewed in this section;
- **Step 2**: Candidate concepts are short-listed;
- **Step 3**: Differences between concepts are reconciled;
- **Step 4**: Chosen concepts are designated into relevant sets (*Mitigation*, *Preparedness*, *Response* and *Recovery*) to facilitate identification of relations between them;
- **Step 5**: Relationships among concepts are identified leading to the initial metamodel;
- **Step 6**: Validating the metamodel.

Existing models used to describe disasters provide a starting point to identify commonly used concepts in DM. However, the models are not comprehensive enough for our purpose as they are too specific to their own context. The first is metamodel of Benaben's [3] expressed using Web Ontology Language (OWL) and focuses on crises management. This metamodel elaborates a common and sharable reference model built to characterize crisis situations in three interrelated views namely *System*, *Treatment System* and *Crisis Description*. Benaben's metamodel covers the whole crisis characterization and collaborative processes that deal with it, aiming to integrate partners through information system interoperability. The second metamodel we use is Kruchten's [5] which conceptualises disasters as encompassing multiple stakeholder domains depicted in four main views: *Disaster Visualization*, *Physical View*, *Communication and Coordination*

Simulator and *Disaster Scenario*. This metamodel attempts to unify the terminology sharpening the definition of terms and their semantic relationships. The third metamodel we consider is Asghar's [4] which focuses on the arrangement of disaster activities in a logical sequence.

Targeting a generic metamodel in our work is inspired by [1] and [2], where a generic metamodel was developed for representing and securing Multi Agent System (MAS). In fact, several generic security concepts identified in [1] have their equivalent in DM. For example, recovering from an intrusion attack in a MAS requires restoring datalogs. Analogies to this exist in restoring many lost community services in disaster scenarios, requiring maintaining back up organizational structures. Our work takes DM modelling a step further aiming to generalize various types of DM activities concepts into one generic encompassing metamodel.

4 DM Metamodel Proposed

Steps 1 to 5 (described in Section 3) are iteratively applied to the metamodels described in the previous section. Our resultant metamodel, the output of steps of 4 and 5, contain the relationships among concepts as shown in Figure 2. It is generic and generalizes various types of disaster concepts that can be refined according to the context on hand. It explicitly covers the management of disaster in all four different phases including *mitigation*, *preparedness*, *response* and *recovery*. We anticipate that various concepts in DM, their relationships and attributes, different types of data models can be generated using refinement of concepts in this metamodel. The core class in this DM Metamodel is the Organisation which represents the loose 'organisation' where DM concepts are operationalised. All key concepts in DM are grouped in the *Organisation* concept. Other key DM concepts are aggregated within this class and they include: *DMProcedure*, *DMRequirement*, *DMPolicy*, *Actor*, *DMTeam*, *DomainKnowledge*, *Resource*, *ActorRole* and *MessageCommunication*. *DMProcedure* can represent the collections of implemented procedures of DM activities including for example Mitigation, Preparedness, Rescue, Response and Evacuation. *DMTeam* defines a collection of *ActorRole* class which typically describes human roles that work towards a *DMGoal*. *ActorTask* class in our metamodel is derived from a *DMGoal* class. Here we also model a *DisasterPreventionGoal* as a class that can be achieved by *DisasterPreventionTask*. Some of the classes from the crisis metamodel developed by Benaben in [3] is taken into consideration while we develop the model of the actual disaster event (left hand side of Figure 2). To model this, we grouped all components consisting of *People*, *Infrastructure*, *NaturalSite* and *CivilianSociety* into one class namely *ElementsAtRisk*. We introduced the '*is a kind of*' (specialization) relationships which tied up these four components with *ElementsAtRisk* class. Thus, *DisasterActionService*, a class of collaboration among several actors will provide support and help to this affected group of elements through the *ElementsAtRisk* class. *Disaster*, a tragedy that affects this *ElementsAtRisk* typically occurs due to accumulation factors represented by a

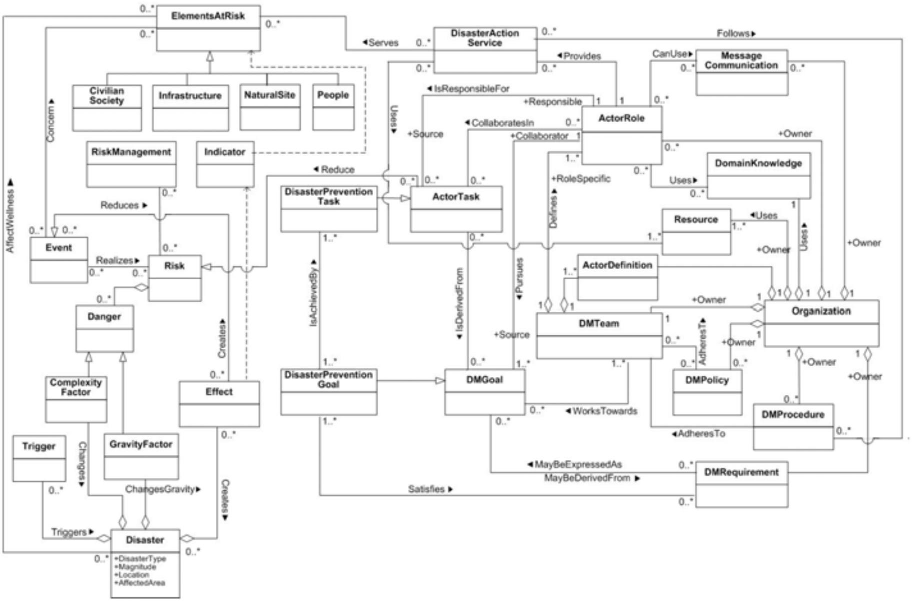


Fig. 2. DM Metamodel proposed

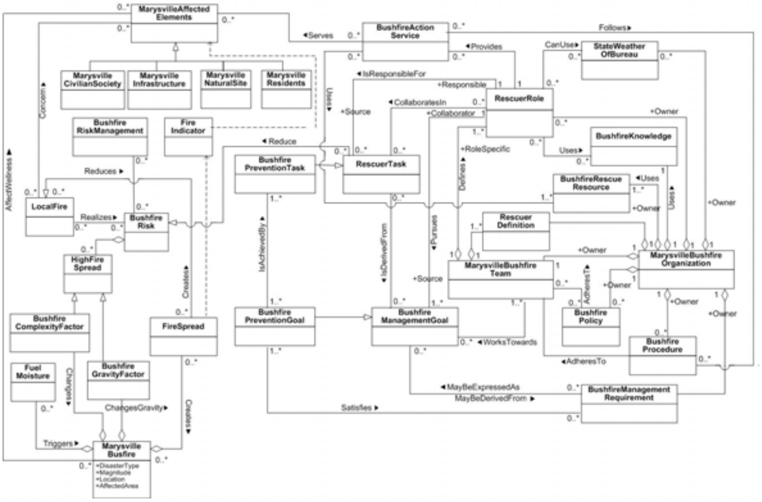


Fig. 3. Metamodel refinement of bushfire disaster in Marysville, Victoria, Australia

Trigger and have consequences that are described by *Effect* and vary in intensity represented by *ComplexityFactor* and *GravityFactor*. Figure 3 represents one of model example that could be derived by using the DM metamodel.

5 Conclusion and Future Work

In this paper, we presented a metamodel (ontology) that will underpin an architecture of a Decision Support System (DSS) for a Generic Disaster Management framework. We presented the DSS architecture where the DM metamodel will be used to represent, store and later retrieve DM knowledge. We outlined and presented the metamodelling process in this domain. As a proof of concept, we presented our first version of a metamodel that can be used to unify DM knowledge. The ability to represent key DM concepts using our metamodel is a preliminary evidence of the feasibility of a generic metamodel in DM. This is unlike most previous attempts in this area which have narrowed their focus on specific types of disasters. We are currently working on a more comprehensive validation which will involve taking 20 existing DM models and ensuring that our metamodel can be refined to generate all of them. Following this validation, we will create a repository of DM knowledge expressed and engineered using our metamodel. In other words, DM actions will be represented as refinement of our metamodel concepts. This will be the first step to develop a DSS which assists in formulating required DM approach based on the disaster event that is provided as input to the system.

References

1. Beydoun, G., Low, G., Mouraditis, H., Henderson-Sellers, B.: A Security-Aware Metamodel For Multi-Agent Systems. Information and Software Technology 51, 832–845 (2008)
2. Beydoun, G., Low, G., Henderson-Sellers, B., Mouraditis, H., Sanz, J.J.G., Pavon, J., Gonzales-Perez, C.: FAML: A Generic Metamodel for MAS Development. IEEE Transactions on Software Engineering 35, 841–863 (2009)
3. Benaben, F., Hanachi, C., Lauras, M., Couget, P., Chapurlat, V.: A Metamodel and its Ontology to Guide Crisis Characterization and its Collaborative Management. In: Proceedings of the 5th International ISCRAM Conference, Washington, USA, pp. 189–196 (2008)
4. Asghar, S., Alahakoon, D., Churilov, L.: A Comprehensive Conceptual Model for Disaster Management. Journal of Humanitarian Assistance (2006)
5. Kruchten, P., Monu, C.W.K., Sotoodeh, M.: A Conceptual Model of Disasters Encompassing Multiple Stakeholder Domains. International Journal of Emergency Management 5, 25–56 (2008)
6. W3C Incubator Group: Emergency Information Interoperability Frameworks, Technical report (2008)
7. Kelly, C.: Simplifying Disasters: Developing a Model For Complex Non-Linear Events Disaster Management. In: Australasia and the Pacific Region Conference, Cairns, Queensland (1998)

Comparison Judgments in Incomplete Saaty Matrices

Henryk Piech and Urszula Bednarska

Czestochowa University of Technology
Dabrowskiego 73, 42-200 Czestochowa, Poland
h.piech@adm.pcz.czest.pl

Abstract. Relative pairwise experts' judgments (assessments) can turn out to be incomplete. Such situation appears, e.g., when experts can not propose precise values representing some judgments. In such situation, Saaty matrices could be completed using well-known averaging methods or the correction method proposed in the current paper. Described method for correction is based on the Consistency Improvement Algorithm.

1 Introduction

In Multiple Criteria Decision Making (MCDM), we need to estimate and correct the values of parameters representing relative judgments. Many methods and their modifications are based on estimating attribute's weights [2,10]. Different variants of these methods have been proposed in literature. Some of them use the eigenvectors and eigenvalues to estimate the inconsistency level [6,13]. There are different algorithms used to calculate eigenvalues, e.g., additional and multiplicative approach. Usually normalized judgments are used. The wide spectrum of methods were used to obtain attribute's weights, e.g., methods based on deterministic [3], interval [2,4], fuzzy [5] and rough sets [1,11] approach. In the interval approach, the statistic experiments [4] or linear programming [7] are used to get interval weights from interval judgment matrix. The method for increasing the consistency level using the correction of judgments is proposed in [8]. In such case, it is required to reach a compromise between experts opinions and the consistency level [9]. The idea, that comes from correcting the given Saaty matrix is used to solve the following problem. The problem is to complete pairwise comparison judgments when the Saaty matrix is incomplete. In order to solve this problem we introduce the relative pairwise judgment matrix characteristics. The proposed method allows achieving judgment consensus step by step (in an iterative process) together with the possibility of investigating convergence to consistency. The algorithm is based on certain features of Saaty matrix and guarantees objective assessment of objects, on the contrary to methods based on subjectively defined classes' thresholds. Involved judgments permit calculating consistency estimator very precisely and unambiguously. Using this approach also relative judgments given by single expert can be corrected. Our

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 622–629, 2010.
© Springer-Verlag Berlin Heidelberg 2010

proposal can give several solutions, but all of them enable us to reach the given consistency level. Incomplete information can be found in different ways (with the help of different reference elements: rows or columns). Such enriched information, in the aspect of reaching the compromise, is comfortable for experts' team, and enables us to estimate expert's competence.

2 The Method for Correction of Relative Deterministic Judgments to Improve the Transitive Profiles of Saaty Matrix $((\boldsymbol{det}(\boldsymbol{A} - \boldsymbol{I\lambda})\boldsymbol{w}) \rightarrow \mathbf{0}$ or $\boldsymbol{\lambda} \rightarrow \boldsymbol{m})$

The proposed method can be presented as follows:
1. Choose the row or column k with the largest credibility (the choice is subjective).
2. Create the correction increments matrix $\mathit{\Delta}$ using the following formulas:
$\mathit{\Delta}(i, j) = a(i, j) - a(k, j)/a(k, i)$ for $k \neq i$
or
$\mathit{\Delta}(i, j) = a(i, j) - a(i, k)/a(j, k)$ for $k \neq j$.
3. Find the maximal absolute value of increment $\mathit{\Delta}_{max}(i_{max}, j_{max})$, where i_{max}, j_{max} – coordinates of the largest discrepancy.
4. Correct relative weights:
$aen(i_{max}, j_{max}) = a(i_{max}, j_{max}) - \mathit{\Delta}_{max}(i_{max}, j_{max})$
5. Calculate the Consistency Index (CI) or the Consistency Ratio (CR) for the obtained matrix.
5.1. Assess the normalized judgments $an(i, j)$ (vector $\mathbf{w}$).
5.2. Calculate vector $\mathbf{u} = \mathbf{A} \cdot \mathbf{w}$.
5.3. Assess the eigenvector λ and its average λ_{aver}.
5.4. Calculate CI and CR.
6. Evaluate requirements concerning consistency (e.g. $CI < 0.01$).
7. Start next iteration from point 1.

The first step of the algorithm generates a problem. The problem is to minimize the subjectivity of selecting the reference row or column k to minimize the number of iterations. One of the simplest solutions of the problem is to estimate the standard deviation in the chosen row or column in reference to the average of all elements above or below the main diagonal:

a) In relation to judgments above the main diagonal, when we choose row vector
$k = \{i; \vartheta rt(i) = \min(\vartheta rt(r); 1 \leq r \leq m - 1)\}$,
$\vartheta rt(i) = \varsigma r(i)/\varsigma top(glob)$,
where
$\varsigma r(i) = 1/(m - i + 1) \sum_{r=i+1}^{m} a(i, r)$,
$\varsigma top(glob) = Aver(glob) = 2(m - 1)/m \sum_{r=1}^{m-1} \sum_{s=r+1}^{m} a(r, s)$.

b) In relation to judgments above the main diagonal, when we choose column vector

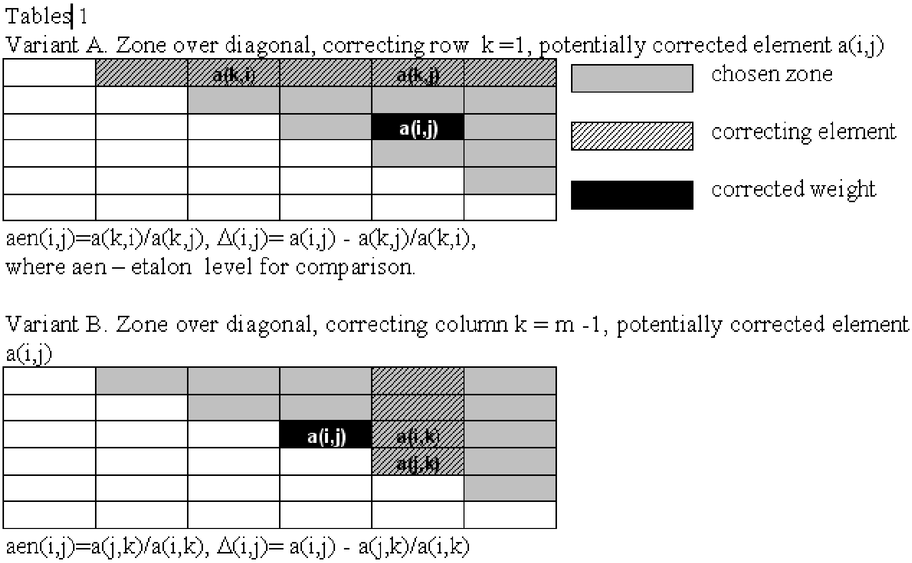
$k = \{i; \vartheta rt(i) = \min(\vartheta rt(r); 2 \leq r \leq m)\}$,
$\vartheta rt(i) = \varsigma r(i)/\varsigma top(glob)$,
where
$\varsigma r(i) = 1/(m-i+1)\sum_{r=i+1}^{m} a(r,i)$,
$\varsigma top(glob) = Aver(glob) = 2(m-1)/m\sum_{r=2}^{m}\sum_{s=i+1}^{m} a(r,s)$.

Similiary, we calculate the average of judgments under the main diagonal.

The above method provides the following preferences: if the element is above the main diagonal then the row number is $k = 1$ and the column number is $c = m$ and if the element is below the main diagonal the $k = m$ and $c = 1$ column. We could consider the uniform reference (when one row or column leads to decreasing the number of iterations and shortening the process) or we can use parts of rows or columns to correct the relative weights. It is convenient to present possible variations graphically.

3 Graphic Presentation of Theversions of the Correction Method

In Table 1, we show some possible variants of choosing correction elements (rows and columns).



Tables 1

Variant A. Zone over diagonal, correcting row k =1, potentially corrected element a(i,j)

	▨	▨ a(k,i)	▨	▨ a(k,j)	▨
		▒	▒	▒	▒
			▒	■ a(i,j)	▒
				▒	▒
					▒

▒ chosen zone

▨ correcting element

■ corrected weight

aen(i,j)=a(k,i)/a(k,j), Δ(i,j)= a(i,j) - a(k,j)/a(k,i),
where aen – etalon level for comparison.

Variant B. Zone over diagonal, correcting column k = m -1, potentially corrected element a(i,j)

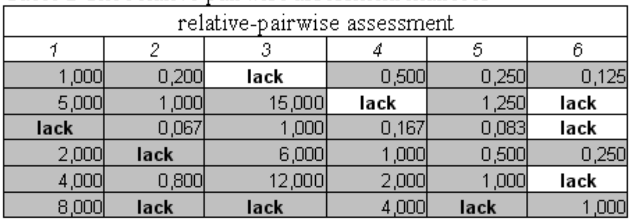
	▒	▒	▒	▨	▒
		▒	▒	▨	▒
			■ a(i,j)	▨ a(i,k)	▒
				▨ a(j,k)	▒
					▒

aen(i,j)=a(j,k)/a(i,k), Δ(i,j)= a(i,j) - a(j,k)/a(i,k)

4 Using the Iterative Correction Method for Completing Saaty Matrices

The relative pairwise comparison judgment matrix created basing on incomplete knowledge has the form presented in Table 2.

Table 2 The relative pairwise assessment matrices



relative-pairwise assessment					
1	2	3	4	5	6
1,000	0,200	**lack**	0,500	0,250	0,125
5,000	1,000	15,000	**lack**	1,250	**lack**
lack	0,067	1,000	0,167	0,083	**lack**
2,000	**lack**	6,000	1,000	0,500	0,250
4,000	0,800	12,000	2,000	1,000	**lack**
8,000	**lack**	**lack**	4,000	**lack**	1,000

In the matrix presented in Table 2 the “lacks” (lack of initial data) can be located in accidental cells. Applying the Correction Method (described below) we can fill the matrix by choosing the row or column vectors. When choosing the correction element we can consider the criteria:

- Minimize the number of corrections,
- Minimize the value of correction increments,
- Minimize the inconsistency (CI),
- Minimize the number of iterations needed to achieve the given consistency level.

In our case, to complete the “lack” in 1-st row vector and 3-rd column ($lack(1, 3)$) we use the correction column $k = 5$:
$lack(1, 3) = a(3, 5)/a(1, 5)$ and $lack(3, 1) = 1/lack(1, 3)$ column $k = 5$
and for other lacks:
$lack(2, 4) = a(4, 5)/a(2, 5)$ and $lack(4, 2) = 1/lack(2, 4)$ column $k = 5$
$lack(2, 6) = a(1, 6)/a(1, 2)$ and $lack(6, 2) = 1/lack(2, 6)$ column $k = 1$
$lack(3, 6) = a(1, 6)/a(1, 3)$ and $lack(6, 3) = 1/lack(3, 6)$ column $k = 1$
$lack(5, 6) = a(1, 6)/a(1, 5)$ and $lack(6, 5) = 1/lack(5, 6)$ column $k = 1$

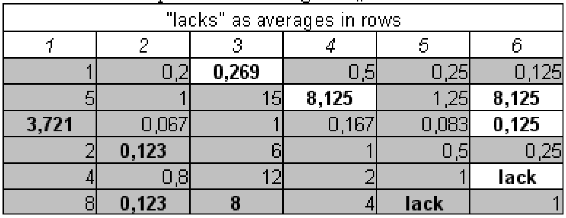
It is also possible to complete the “lack” using another method, which is based on the average of elements in the row or column (only elements above or below the main diagonal are considered):
$lack(i, j) = 1/(m - i - nwl) \sum_{r=i+1; r\neq j; a(i,r)\neq \text{“}lack\text{“}}^{m} a(i, r)$ (row over diagonal)
or
$lack(i, j) = 1/(j - 1 - nkl) \sum_{s=1; s\neq j; a(s,j)\neq \text{“}lack\text{“}}^{j-1} a(s, j)$ (column under diagonal)
where
nwl – the number of “lacks” in the row chosen,
nkl – the number of “lacks” in the column chosen.
For example:
$lack(1, 3) = 1/(6 - 1 - 1) \sum_{r=2; r\neq 4;}^{6} a(1, r)$
or
$lack(1, 3) = 1/(3 - 1 - 1) \sum_{s=1; s\neq 1;}^{2} a(s, 3)$.

When choosing the rows or columns for averages we consider the following factors:

- Situation structure (location of “lacks” in the matrix),
- The amount of data in row or column.

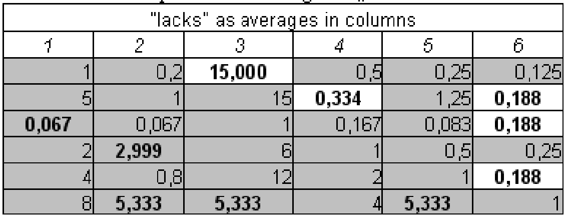
Using both methods we obtain matrices presented in Table 3(a),(b).

Table 3a Example of fulfilling the „lacks"



"lacks" as averages in rows					
1	*2*	*3*	*4*	*5*	*6*
1	0,2	**0,269**	0,5	0,25	0,125
5	1	15	**8,125**	1,25	**8,125**
3,721	0,067	1	0,167	0,083	**0,125**
2	**0,123**	6	1	0,5	0,25
4	0,8	12	2	1	**lack**
8	**0,123**	**8**	4	**lack**	1

Table 3b Example of fulfilling the „lacks"



"lacks" as averages in columns					
1	*2*	*3*	*4*	*5*	*6*
1	0,2	**15,000**	0,5	0,25	0,125
5	1	15	**0,334**	1,25	**0,188**
0,067	0,067	1	0,167	0,083	**0,188**
2	**2,999**	6	1	0,5	0,25
4	0,8	12	2	1	**0,188**
8	**5,333**	**5,333**	4	**5,333**	1

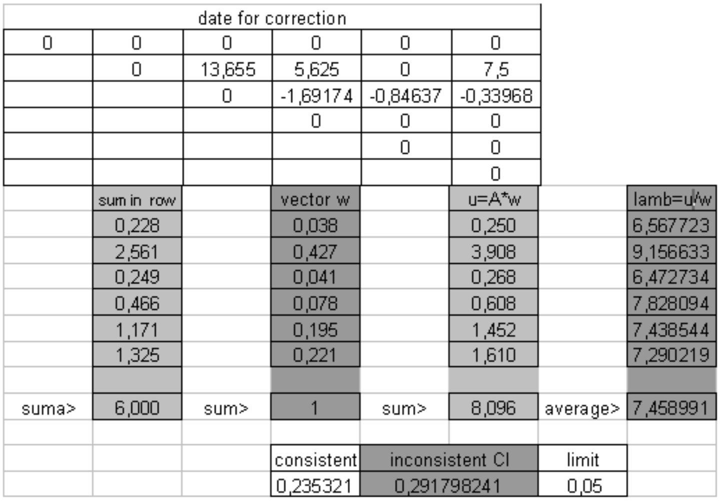
5 Experiments and Results

The experiment is based on data from Table 3. In matrix 3a we completed the 'lack' using the formulae: $a(5,6) = a(4,6)/a(4,5)$.

We analysed the data according to the algorithm which was presented in first section and stored the results in Table 4.

Tables 4 Iteration process.

iteration 1 (for date from tabl. 3a)



date for correction							
0	0	0	0	0	0		
	0	13,655	5,625	0	7,5		
		0	-1,69174	-0,84637	-0,33968		
			0	0	0		
				0	0		
					0		
	sum in row		vector w		u=A*w		lamb=u/w
	0,228		0,038		0,250		6,567723
	2,561		0,427		3,908		9,156633
	0,249		0,041		0,268		6,472734
	0,466		0,078		0,608		7,828094
	1,171		0,195		1,452		7,438544
	1,325		0,221		1,610		7,290219
suma>	6,000	sum>	1	sum>	8,096	average>	7,458991
			consistent	inconsistent CI		limit	
			0,235321	0,291798241		0,05	

For data from Table 3(a) we need 5 iterations ($consistency \leq limit$). Finaly we obtain consistent level showed in Table 5.

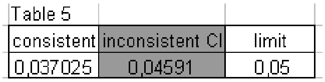


Table 5

consistent	inconsistent CI	limit
0,037025	0,04591	0,05

For data from Table 3(b) we needed 9 iterations. The final consistency level is presented in Table 6.

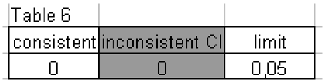


Table 6

consistent	inconsistent CI	limit
0	0	0,05

The maximum number of iterations for the given $m \times m$ matrix is $m(m-1)/2 = 6 \cdot 5/2 = 15$.

6 Averaged Correction Method

The assumptions of the Iterative Correction Method show that correction elements can be rows (columns) which numbers are lower (greater) than the number of the corrected row (column) (Tables 7, 8).

Table 7. Direction of influence correcting rows

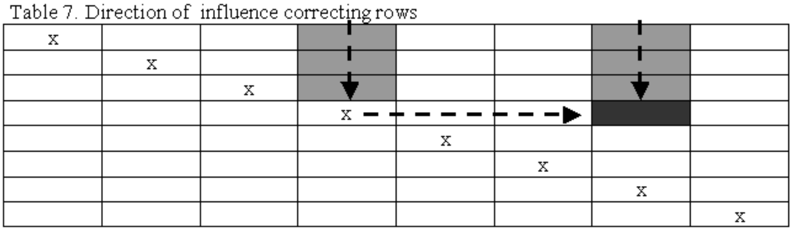
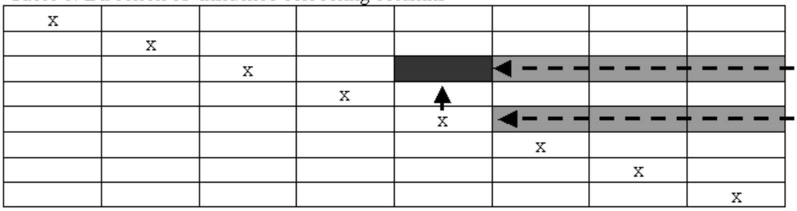


Table 8. Direction of influence correcting columns



$a^{(1)}(4,7) = a^{(0)}(1,7)/a^{(0)}(1,4)$,
$a^{(2)}(4,7) = a^{(0)}(2,7)/a^{(0)}(2,4)$,
$a^{(3)}(4,7) = a^{(0)}(3,7)/a^{(0)}(3,4)$,
where
$a^{(r)}(i,j)$ – element in i-th row and j-th column which was corrected by r-th row $(r < i)$

$a^{(8)}(3,5) = a^{(0)}(3,8)/a^{(0)}(5,8)$,
$a^{(7)}(3,5) = a^{(0)}(3,7)/a^{(0)}(5,7)$,
$a^{(6)}(3,5) = a^{(0)}(3,6)/a^{(0)}(5,6)$,
where
$a^{(c)}(i,j)$ – element in i-th row and j-th column corrected by c-th column $(c > j)$.

The proposed variation of the method is designed to eliminate the subjective character of the Correction Method and therefore eliminate the problem which is described in Section 2 of this paper. The proposed Averaged Method is based on general formulas:
$a^{(1)}(i,j) = a^{(0)}(1,j)/a^{(0)}(1,i)$,
$a^{(2)}(i,j) = a^{(0)}(2,j)/a^{(0)}(2,i)$,
...
$a^{(i-1)}(i,j) = a^{(0)}(i-1,j)/a^{(0)}(i-1,i)$
or
$a^{(m)}(i,j) = a^{(0)}(i,m)/a^{(0)}(j,m)$,
$a^{(m-1)}(i,j) = a^{(0)}(i,m-1)/a^{(0)}(j,m-1)$,
...
$a^{(j+1)}(i,j) = a^{(0)}(i,j+1)/a^{(0)}(j,j+1)$.

The idea of the method is to use averaged relative judgments. Instead of using only one selected element, we use averages of elements in rows and columns:
$a'(i,j) = 1/(i-1)\sum_{p=1}^{i-1} a^0(p,j)a^0(p,i)$,
$a''(i,j) = 1/(m-j)\sum_{p=j+1}^{m} a^0(i,p)a^0(j,p)$
or
$a'''(i,j) = 1/(m-j+i-1)(\sum_{p=1}^{i-1} a^0(p,j)a^0(p,i) + \sum_{p=j+1}^{m} a^0(i,p)a^0(j,p))$.

Estimation of correction increments using averaged correction elements:
$\Delta(i,j) = a^{(0)}(i,j) - a'(i,j)$,
$\Delta(i,j) = a^{(0)}(i,j) - a''(i,j)$
or
$\Delta(i,j) = a^{(0)}(i,j) - a'''(i,j)$.

This variant of the basic method is more objective as it uses all elements, not only the one which is subjectively chosen.

7 Conclusions

When dealing with incomplete data, we have several possibilities of using different properties of relative pairwise comparison matrices. After creating an option estimator the next sphere for applying it appeared and the possibility of completing the Saaty matrix increased. When using the proposed methods for completing the relative matrix, we simultaneously improve the consistency level. An iterative character of these methods gives us additional chance to use different, just created, new elements in the next iterations. Generally, we always obtain more consistent matrices, with better characteristics than, when using other methods.

References

1. Dong, Y., Xu, Y., Li, H.: On consistency measures of linguistic preference relation. European Journal of Operational Research 189, 430–444 (2008)
2. Kazumoti, S., Hiroaki, I., Hideo, T.: Interval priorities by interval regression analysis. European Journal of Operational Research 158, 745–754 (2004)
3. Krawczyk, S.: Metody ilościowe w planowaniu działalności przedsiebiorstwa. C.H. Beck, Warszawa (2001)
4. Mikhailov, L.: Deriving priorities from fuzzy pairwise comparison judgments. Fuzzy sets and systems 134, 365–385 (2003)
5. Mikhailov, L.: Group prioritization in the AHP by fuzzy preference programming method. Computer and Operation Research 31, 293–301 (2004)
6. Piech, H.: The methodology of improvement of consistent in Saaty's matrix judgments. Scientific Research of the Institute of Mathematics and Computer Science 1(7), 159–170 (2008)
7. Saaty, T.L.: Fundamentals of decision Making and Priority Theory with the Analytic Hierarchy Process. RWS Publications, Pittsburgh (1994)
8. Saaty, T.L.: The Analitic Hierarchy Process. McGrow-Hill, New York (1980)
9. Saaty, T.L.: Multicriteria Decision Making: Analytic Hierarchy Process. RWS Publications, Pittsburgh (1994)
10. Sharma, J.M., Moon, I., Bae, H.: Analytic hierarchy process to assess and optimize distribution network. Applied Mathematics and Computation 202, 256–265 (2008)
11. Walley, P.: Belief - function representations of statistical evidence. Ann. Stat., 741–761 (1987)
12. Wang, Y.M., Elhag, T.M.S., Hua, Z.S.: A modified fuzzy logarithmic least squares method for fuzzy analytic hierarchy process. Fuzzy Sets and Systems 157, 3055–3071 (2006)
13. Wang, Y.M., Parkan, C., Luo, Y.: A linear programming method for generating the most favorable weights from a pairwise comparison matrix. Computers and Operational Research 35, 3918–3930 (2008)

Application of an Expert System for Some Logistic Problems

Andrzej Pieczyński[1] and Silva Robak[2]

[1] Institute of Control and Computation Engineering
[2] Faculty of Mathematics, Computer Science and Econometrics
The University of Zielona Góra
ul. Podgórna 50, 65-246 Zielona Góra, Poland
A.Pieczynski@issi.uz.zgora.pl,
S.Robak@wmie.uz.zgora.pl
http://www.uz.zgora.pl

Abstract. In the paper some problems associated with the application of an expert system for a solution of some logistic problems will be considered. The Decision Support System (DSS) that condenses large amount of data is applied to support the managers who develop contracts to carry bulk cargos from the company warehouses to the customers. Because of the fact that some of the transportation contracts' characteristics are imprecise or unpredictable, the authors suggested their descriptions on the basis of fuzzy logic and probability methods. The analytical power of the system will be supported by the fuzzy expert system enabling the choice of the appropriate customer orders to maximize the company's profits. The considered decision-support system has been implemented and validated in the environment of the Exsys expert system.

Keywords: business intelligence, decision-support systems, heuristic decision tree, fuzzy logic, probability, expert systems, Exsys.

1 Introduction and the Problem

In the contemporary world, information systems have gained increased contribution in their usefulness to business organizations. More specifically, the change comes from extended opportunities they give in management decision-making process.

Decision-support Systems (DSS), as "business intelligence"-systems, are of great value in improved decision-making process for senior and middle management, operational management, and also individual employees and project teams. The decision characteristics may vary, depending on decision-making level. High quality decision-making is especially important in unstructured and semi-structured decisions. There are certain stages in decision making such as intelligent design, choice and implementation. Moreover, there are different types of DSS, basically there are model-driven or data-driven systems. The model-driven systems have good analytical capabilities based on theory and analytical

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 630–637, 2010.

© Springer-Verlag Berlin Heidelberg 2010

models and usually they include user-friendly graphical user interfaces. The data-driven systems are capable of analysis of large pools of data, often collected from transactions processing systems in data warehouses or data marts. DSS enable carrying out analysis by application of libraries of diverse analytical models such as statistical models, optimization models and forecasting models. They also provide means for accomplishment of the "what-if"-sensitivity analysis. In case of the problems with poor quality of analytical models, we may complement them with the heuristic data mining methods [6].

In this paper we consider a selection of diverse solution alternatives for a suitable choice of customers with a voyage-estimating program for a bulk transport. It will be aided with fuzzy logic and probability methods applied in the modules of the Exsys expert system developed at the University of Zielona Gora.

The further contents of our paper is structured as follows. In Section 2 we present considerations on the estimating program for managers who plan and develop transport contracts for a company. For bulk transport with trucks (from company warehouses to customers), motivation and a choice of features in transportation planning will be considered. Section 3 contains some considerations about the fuzzy set theory and implementation of our fuzzy expert system. In Section 4 we conclude our work.

2 A DSS for Managers Developing Contracts for Bulk Transport with Trucks

In the study, we analyze the instance of a company planning transport contracts with own trucks, but when needed, the firm may also lease some external transportation means. A DSS as an interactive system that operates on a powerful desktop computer provides a system of menus that makes it easy for its users to input data or obtain information. It may aid the managers with its analysis of financial and technical aspects of transport contracts. Technical aspects in the voyage estimation include such variables as: the distances between the company warehouses and customers, the cargo capacity of the transport means, speed, place distances, fuel consumption, the load patterns, etc. The listed technical details have rather a stable, invariant character. The financial aspects include costs of the transportation means and time, fright rates for different types of cargo and other expenses. The first mentioned financial factor, which includes the costs of fuel and labor, may vary because of some dependencies on weather conditions, and the time spent waiting for unloading/loading, etc.

The DSS would work in an interactive way - the user can change assumptions, ask new questions and input new data. The analytical model database collects and stores knowledge of the system based on files like a truck file (speed, capacity, hire history cost file), a distances file, a fuel consumption cost file, and a customer data file. Special software may help to determine value, revenue potential and loyalty of each customer in order to help managers to make better decisions in developing transportation contracts. In order to achieve that the system may segment customers into several categories based e.g. on their needs, attitudes, and behaviors or significance of the client for the company.

In the next Section some basic considerations about fuzzy sets will be presented, they will be applied in modeling of imprecise features of bulk transportation with trucks.

3 Application of Fuzzy Expert System for Decision Support in Bulk Transportation

Decision support system will generate a suggestion on how to select a client for cargo transportation between a company warehouse and a client. The expert system will deduct, having analyzed the following voyage features:

□ Transportation cost defined as aggregation of a distance between a customer and a warehouse, fuel consumption of a transportation mean (a truck) and possible obstacles expected during a planned voyage (probability of the opposite wind);

□ Order capacity value defined as aggregation of order capacity declared on paper (a hardcopy order that is sent from customer to a warehouse office) and a kind of client regarding its importance for the company;

□ Warehouse bulk resources;

□ Warehouse owner profit defined as aggregation of a client's profile and transportation cost.

3.1 Knowledge Base of the Fuzzy Expert System

Most of the analyzed features will be defined with description based on fuzzy sets. A few of the above mentioned features are continuous variables. The probability functions are discrete variables [8]. The membership functions [1],[4] used for all described features are shown in the Fig. 1.

The distance between a customer and a warehouse, order capacity, truck fuel consumption, transport cost have been described by means of three fuzzy sets: low, medium, high (see Fig. 1.a). The client profile variable has been described also by means of three fuzzy sets namely: new, stable and very important (see Fig. 1.b). Two fuzzy sets (empty and full) were used to describe the embarkation rate of truck (see Fig. 1.c). The warehouse owner profit has been defined with five fuzzy sets: very low, low, medium, high and very high (see Fig. 1.d). Transportation conditions have been described by means of three fuzzy sets: very bad, bad, good (see Fig. 1.f).

Aggregation mechanism is presented below as formulas (1) - (3). The order capacity value may be described in the following way [2],[5]:

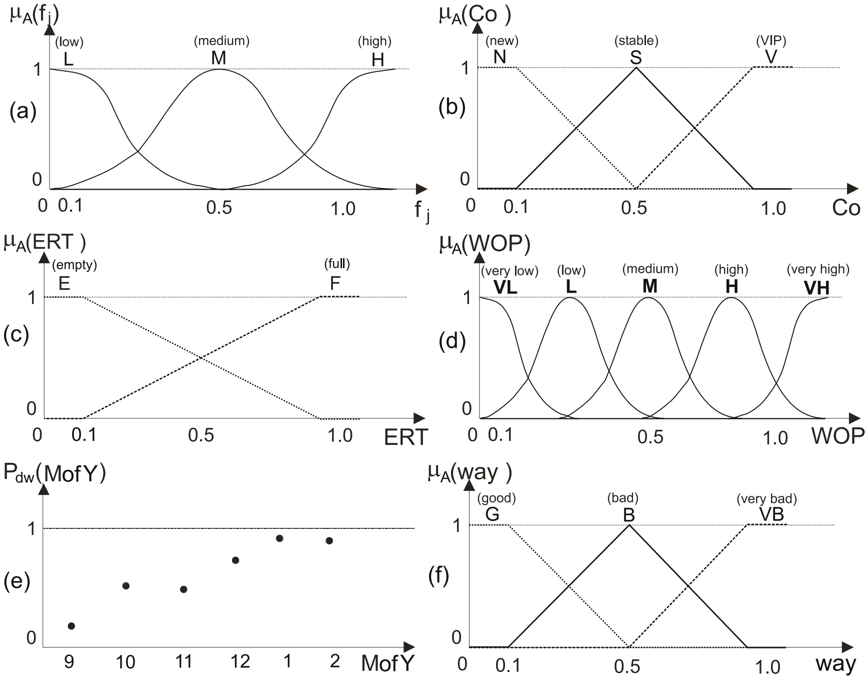


Fig. 1. Membership functions used for representation of all fuzzy features

$$OC(POC, W_{POC}, PC, W_{PC}) = \max\left\{\left[\sum_{i=L}^{H} \mu_i(POC)\cdot w_{POC_i}\right], \left[\sum_{i=L}^{H} \mu_i(PC)\cdot w_{PC_i}\right]\right\}. \tag{1}$$

where: POC is the paper order capacity, W_{POC} is the vector of the fuzzy capacity value influence on the customer importance, PC is the profile of a customer, and W_{PC} is the customer importance factor.

The embarkation rate of truck, distance between the warehouse and the customer location and the fuel consumption of trucks are described in the following equation:

$$TC(ERT, W_{ERT}, D, W_D, FC, W_{FC}) = \left[1 - \sum_{i=E}^{F} \mu_i\left(ERT\right)\cdot w_{ERT_i}\right] \cup \left[\sum_{i=L}^{H} \mu_i\left(D\right)\cdot w_{D_i}\right] \cup \left[\sum_{i=N}^{V} \mu_i\left(FC\right)\cdot w_{FC_i}\right]. \tag{2}$$

where: TC is transport cost, ERT is embarkation rate of a truck, W_{ERT} is embarkation rate of a truck influence on the transport cost, D is distance between a company warehouse and a customer location, FC is truck fuel consumption, W_D and W_{FC} are the distance and the fuel consumption influences on transport cost.

Trucks fuel consumption is described in the following way:

$$FC(AFC, W_{AFC}, KW, W_{KW}, P_{dw}) = \left\{ \left[\sum_{i=L}^{H} \mu_i(AFC) \cdot w_{AFC_i} \right] \cap \left[\sum_{i=G}^{VB} \mu_i(KW) \cdot w_{KW_i} \right] \right\} \cdot P_{dw}. \tag{3}$$

where: FC is fuel consumption, AFC is average fuel consumption of truck, KW - are transportation conditions, W_{AFC} and W_{KW} are fuzzy value importance vectors of the average fuel consumption and transportation conditions, P_{dw} is the probability of bad weather.

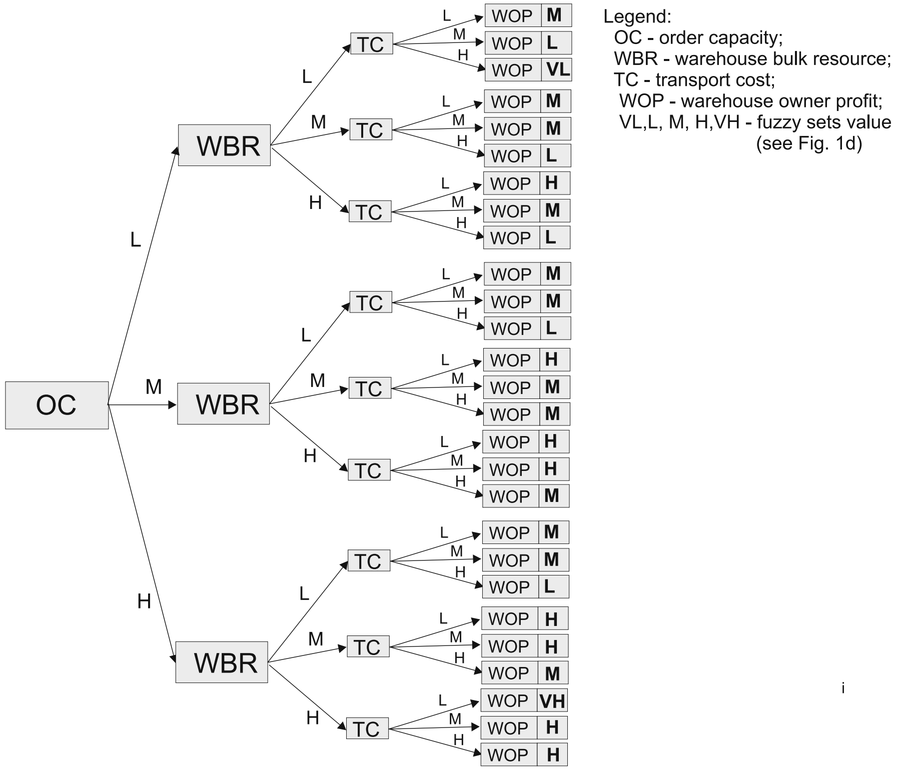
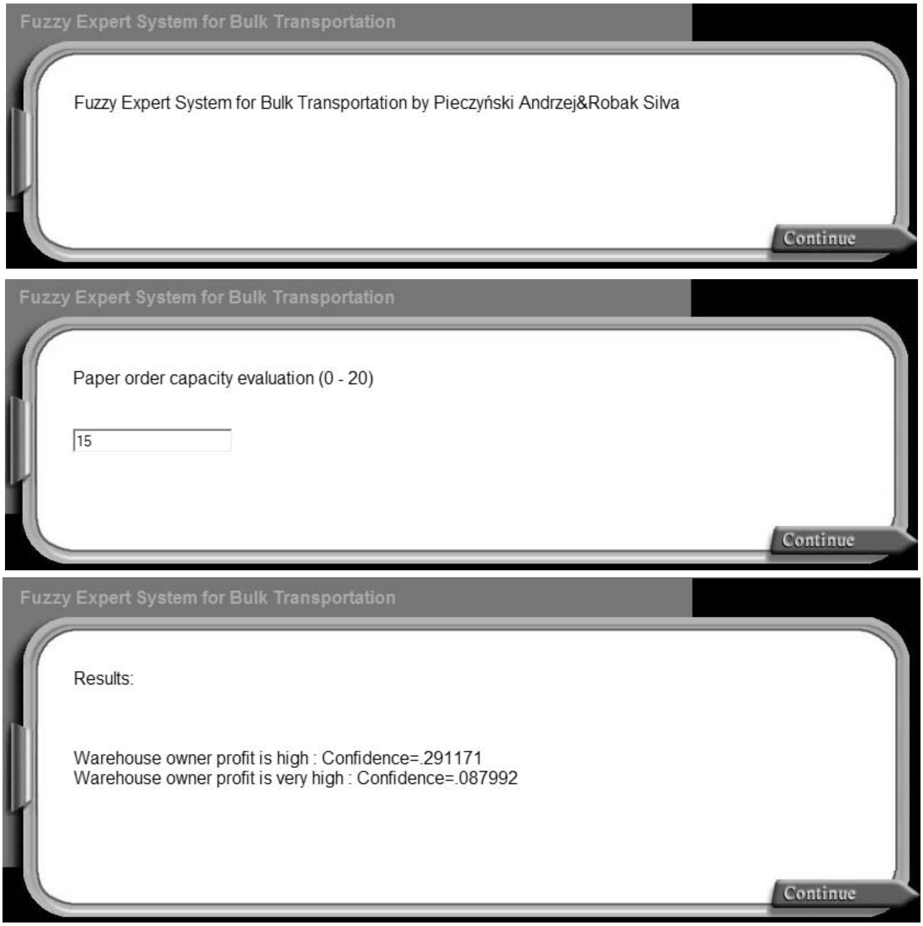


Fig. 2. The decision tree in the fuzzy expert system

The decision tree [3], which describes knowledge base used in our system, is shown in the Fig. 2.

3.2 Implementation of the Fuzzy Expert System

The proposed solution was implemented in the expert shell system EXSYS Developer. The knowledge base for this system shown in Fig. 2 is the basis for automated generation of the knowledge rules. Such kind of knowledge representation is advantageous because of many possible hierarchies of the conclusions with diverse certainty levels. The conception of a rule established knowledge base, allows for acquisition of conclusion sets with assigned certainty factors.
For instance:
IF *order capacity* is **L** and *warehouse bulk resource* is **L** and *transport cost* is **L** **THEN** *warehouse owner profit is medium*
IF *order capacity* is **M** and *warehouse bulk resource* is **M** and *transport cost* is **L THEN** *warehouse owner profit is high*
....


Fuzzy Expert System for Bulk Transportation
Continue
Fuzzy Expert System for Bulk Transportation
Continue
Fuzzy Expert System for Bulk Transportation
Continue

Fuzzy Expert System for Bulk Transportation by Pieczyński Andrzej&Robak Silva

Paper order capacity evaluation (0 - 20)

15

Results:

Warehouse owner profit is high : Confidence=.291171
Warehouse owner profit is very high : Confidence=.087992

Fig. 3. The fuzzy expert system answer - a screen dump

IF *order capacity* is **H** and *warehouse bulk resource* is **L** and *transport cost* is **H** **THEN** *warehouse owner profit is low*

The applied shell expert system provides the needed consistency checks on the knowledge base. In this version, the tree representation uses 45 nodes. The user interface construction was based on special script language, implemented in the EXSYS shell program (see Fig. 3).

4 Conclusion

The proposed system supports decision making process facilitating the selection of appropriate transportation offers. In the system, in the fuzzy representation, such aspects as: the place distance, costs, load volume of the trucks, difficulties during the transportation have been taken into account.

The fuzzy representation of knowledge made it available to construct a transparent knowledge base and a flexible inference system.

The implemented solution allowed us to validate the anticipated thesis, as shown in our paper on a simple example. The used shell expert system Exsys facilitates rapid development of an expert system. Nevertheless, one serious disadvantage of the implemented system could be the high cost of Exsys, thus not applicable for small expert systems. Therefore, for small expert systems development, application of a dedicated system may be a better solution.

In the paper proposition of full decision tree containing all possible combinations has been presented, which results in non-optimal and redundant tree structure. In the further work, optimization of a decision tree, for instance with the Quinlann method, should follow.

The knowledge representation in case of more complex problem domains may be extended with some additional models, like feature diagrams as shown in [11],[12],[13],[14]. The feature diagrams may be modelled with standard UML tools as suggested in [10].

Similar work considering the decision-support systems and fuzzy modelling of some system features in the domain of web services have been presented in [15],[16].

The expert systems with usage of the fuzzy set theory and fuzzy logic have been successfully applied in the domain of the medicine. The expert systems support i.e. the reasoning in medical diagnosis and treatment [7] and quick decision-making with a real-time system applied in the intensive care unit [17].

References

1. Dubois, F., Hullermeister, D., Prade, H.: Computer Science Today. Fuzzy Set-Based Methods in Instance-Based Reasoning. IEEE Trans. on Fuzzy Systems 10(3), 322–332 (2002)
2. Fedrizzi, M., Kacprzyk, J.: Brief introduction to fuzzy sets and fuzzy arithmetic. In: Kacprzyk, J., Fedrizzi, M. (eds.) Studies on Fuzziness, vol. 1. Springer, Heidelberg (1992)

3. Kwiatkowska, A.M.: Decision support systems. How use knowledge and information, in practise. PWN, Warsaw (2007) (in Polish)
4. Leski, J.: Neuro-Fuzzy Systems. WNT, Warsaw (2008) (in Polish)
5. Kosinski, W.: On fuzzy number calculus. International Journal of Applied Mathematics and Computer Sciences 16(1), 51–57 (2006)
6. Kulczycki, P., Hryniewicz, O., Kacprzyk, J. (eds.): Information Technology in Systems Studies. WNT, Warsaw (2007) (in Polish)
7. Nguyen, H.: Fuzzy set theory and medical expert systems: Survey and model. In: Bartosek, M., Staudek, J., Wiedermann, J. (eds.) SOFSEM 1995. LNCS, vol. 1012, pp. 431–436. Springer, Heidelberg (1995)
8. Pieczynski, A., Robak, S., Walaszek Babiszewska, A.: Features with fuzzy probability. In: Proc. 11th IEEE International Conference on Engineering of Computer-Based Systems, ECBS 2004. IEEE, Los Alamitos (2004)
9. Robak, S.: Contribution to the improvement of the software development process for product families. Monography, Uniwersity of Zielna Gora Press, Zielona Gora (2006)
10. Robak, S., Franczyk, B., Politowicz, K.: Extending the UML for modeling variability for system families. International Journal of Applied Mathematics and Computer Science 12(2), 285–298 (2002)
11. Robak, S., Pieczynski, A.: Employing fuzzy logic in feature diagrams to model variability in software product-lines. In: Proc. 10th IEEE International Conference on Engineering of Computer-Based Systemsm, ECBS 2003, Huntsville Alabama. IEEE, Los Alamitos (2003)
12. Robak, S., Pieczynski, A.: Employment of Fuzzy Logic in Feature Diagrams to Model Variability in Software Families. Journal of Integrated Design and Process Science 7(3), 79–94 (2003)
13. Robak, S., Pieczynski, A.: Application of fuzzy weighted feature diagrams to model variability in software families. In: Rutkowski, L., Siekmann, J.H., Tadeusiewicz, R., Zadeh, L.A. (eds.) ICAISC 2004. LNCS (LNAI), vol. 3070, pp. 370–375. Springer, Heidelberg (2004)
14. Robak, S., Pieczynski, A.: Adjusting software-intensive systems developed by using software factories and fuzzy features. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 297–305. Springer, Heidelberg (2006)
15. Robak, S., Pieczynski, A.: Fuzzy modeling of QoS for e-business transaction realized by web services. J. of Applied Computer Science. 16, 69–79 (2008)
16. Robak, S., Pieczynski, A.: Decision-support system for the maritime trade. Polish Journal of Environmental Studies 18(4B), 172–176 (2009)
17. Schuh, C.: Managing Uncertainty with Fuzzy-Automata and Control in an Intensive Care Environment. Advances in Soft Computing, vol. 42, pp. 263–271. Springer, Heidelberg (2007)

AI Methods for a Prediction of the Pedagogical Efficiency Factors for Classical and e-Learning System

Krzysztof Przybyszewski

IT Institute
Department of International Studies and Computer Science
Academy of Management in Lodz (SWSPiZ), Sienkiewicza 9, Lodz, Poland
kprzybyszewski@swspiz.pl

Abstract. The idea to apply the selected AI methods for the determination and prediction of the pedagogical efficiency of the classical and e-learning systems have been described in the paper. The partial and total information functions have been defined for such systems treated like the information systems. The values of the partial information function for the system elements or granules of them are the pedagogical efficiency factors and it is possible to use them for the prediction of the total pedagogical efficiency factor of systems. It is possible to do a prediction only by the AI methods.

1 Introduction

It is possible to treat each learning system like an information system. That applies to the classical learning system and to the computer/internet supported learning (e-learning, distance learning, intelligent tutoring) systems, as well [1,2]. There is one general assignment for each learning system: to develop the suitable competences (the skills based on the exact, structural and oriented knowledge) in each participant (student). It is one way to determine an efficiency of such systems: we have to estimate the relations (ratios) between students achievement and organizers expenses (time of learning, the costs of aids and technical equipment, etc.).

We can define efficiency of each learning system (or the elements of it) by determination of a set of the parameters or the values of an exact function. It is similar to a definition of an information function for a learning system treated like an information system [2].

According to the definition of an information system [3,4,5], an information function is one of ways to present an information about the elements of that system, which are characterized with one set of the similar features. A set of the features is depended on the evaluation goals. The values of an information function are useful for their taxonomy of the system elements or for a prediction of the efficiency factors of ones. An example of the prediction of the economical efficiency factor for the learning system based on suitable construction of an information function, has been presented in Kruss paper [6].

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 638–644, 2010.
© Springer-Verlag Berlin Heidelberg 2010

A pedagogical efficiency factor for learning system is defined most often. It ought to determine a rank of the learning results (based on such parameters as: the marks, the a factor of employment of graduates, the results of professional qualifying exams or the factor of a workstation standard conformity [7]) accommodation to the requirements (employers or higher level schools requirements) or to the standards which have been worked out before (e.g. the ECTS system that has been worked out in the Bologna process[1]).

2 Application of the Learning System Information Functions for Its Pedagogical Efficiency Factor Determination

We can determine efficiency factor of some various aspects of the learning systems. A unit system student-teacher is the crux (the elementary component) of each learning system, even the most complex one. A schema of such component that is based on a cybernetic model of a learning process [8] is presented on Fig.1:

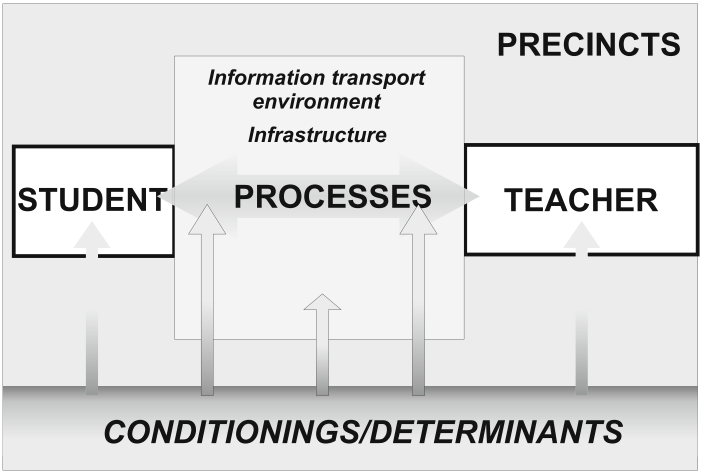


Fig. 1. The elementary component of a learning system schema

The elements of that elementary components have been granulated (i.e. students have been joined into teams, the similar problems have been integrated into subjects), but they have been dispersed (i.e. one can learn some different field of study an interdisciplinary studies that are taken place in many centers or schools) and they have been made virtual (that is defined as replacing of

[1] www.bologna-bergen2005.no
www.dfes.gov.uk/bologna

the real states and stimulus with the simulations that affect directly on human receptors).

In that case, the separate elements of the elementary components: student, teacher, information transport environment, infrastructure and even processes (ways of realization) are evaluated. It is possible to evaluate an organization of elementary component and an influence of the precincts (the determinants) on a system (the elements of it) and on the processes proceeded inside it.

If we treat the learning system like the information system we are able to use the system analysis methods (they seem to be the best [2]) and we ought to separate all system components:

- the system elements,
- a infrastructure (or the information transport environment),
- the processes,
- the precincts (the determinants).

In order to determine the efficiency factor of any learning system treated like information system (or elements of it) we ought to define an information function in a suitable domain (a set of the elements features or of the processes features or a set of the determinants). The separation of the system elements and the processes and the precincts enables a determination of the independent information functions for the separated objects of a system (they are called partial information function). It is possible to do conditional evaluation that is limited to the similar objects of a system. We can determine an efficiency factor of whole system (make total evaluation) by calculating the convolution of the partial information functions.

In a case of the pedagogical efficiency factor of a learning system, the accepted marking scale (grading system) is a set of the values of an information function and it can be represented in the different ways [9]. The representation of a grading system based on the fuzzy numbers seems to be the best, because it enable to determine an efficiency factor of a learning process for an individual student [10], for the student groups and it enable to compare the effectiveness of the teachers [1].

The most often it is carried out a conditional evaluation of the learning systems built of the granules of the elements (i.e. schools which contain the groups of students and the groups of the teachers, or the school district which is a set of schools). In that case it is easy to describe a conditional evaluation procedure based on a suitable standardized information function for the students or teachers teams [1,2,11]. It is example of the evaluation oriented at a student (or teacher) that determine a pedagogical efficiency factor of a learning system.

The description of a comparative evaluation of the pedagogical efficiency factors for schools in the districts in Poland based on an estimation of a educational added value is contained in Pokropek's publication [12]. Another method of an estimation of a pedagogical efficiency for one gymnasium have been described at my earlier paper [13].

3 Application of the Learning System Information Functions for a Prediction of Its Pedagogical Efficiency Factor

We can predict a pedagogical efficiency of a learning system from the known, standardized results for earlier time periods. The results standardization means to use the standardized grading systems. The standard nine (stanine) scale [2], [14] seems to be the best. It is a method of scaling scores (marks) on a nine-point standard scale with a mean of five and a standard deviation of two. In this case, a set of the information function values contains nine elements and it is possible to represent them by a linguistic representation SMS_{ling}:

the lowest results, very low results, low results, below average results,
average results, above average results, high results, very high results,
the highest results

or by the numerical representation SMS_{num}:

$$SMS_{num} = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

or by the fuzzy representation (representation in the trapezoid fuzzy numbers set – $SMS_{fuz)}$ [1]):

$$SMS_{fuz} = \{[0; 0; 1,5; 9], [0; 1,5; 2,5; 9], ..., [0; 8,5; 9; 9]\}$$

The determination of the expected final (outline) values of a pedagogical efficiency factors for each elementary component (student) (the outline values of a partial information function yi) is the aim of a prediction process. Those values are determined on the basis of the known results of students achievements for an earlier periods that are the initial partial information function values (ui) and the known parameters of determinants (i.e. specific values of a partial information function for precincts zi). The forecast for a compound learning system can be generated by estimation of an educational added value per one student on the basis of the prediction of their pedagogical efficiency factors. The prediction process of a pedagogical efficiency factor for individual student is presented in Fig.2.

It is possible to carried out the prediction process in different ways:

1. The efficiency factor values are predicted from the classical statistical estimation methods (linear or non-linear);
2. The efficiency factor values are predicted by application of the estimation on a fuzzy representation of the input values;
3. The efficiency factor values are predicted by application of an artificial neural network based on a classical (linguistic or numerical) representation of the input values;
4. The efficiency factor values are predicted by application of an artificial neural network based on a fuzzy representation of the input values (or an artificial neural network with fuzzy artificial neurons).

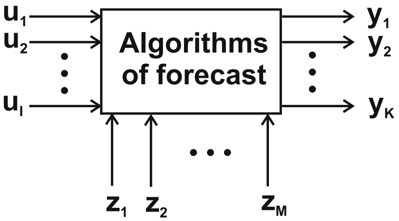


Fig. 2. The elementary component of a learning system schema

4 The Results of the Pedagogical Efficiency Factors Prediction for the Selected Gymnasium

The results of the final exams that will be carried out on April 2010 and the pedagogical efficiency factors corresponded to them, have been predicted for selected gymnasium (Public Gymnasium no 26 in Lodz). The values of stanin scale calculated from the results of the competence test carried out for the end of a primary school for each student at 2007 (u_1) and an average of her/his marks (grades) at the same moment (u_2) and her/his primary school number (as the values of a partial information function for precincts, z_1), have been the input values. The forecast have been based on the known results for 2009 and 2008 and 2007, because only those data were available. The forecasting have been done for two exams: a humanistic one (arts) and a scientific/nature one (the output values: y_1 and y_2).

The results obtained with methods 1 and 2 have been calculated separately for each primary school that students came into the gymnasium three years before. It is right way for taking into consideration influence of the precincts parameters.

The results obtained with method 1 have not been precise in statistical sense: the values of correlation coefficient were smaller then 0,6 for the linear estimation and they were smaller then 0,5 for exponential one. It is a normal situation because the learning process is not a chance process. That method is not useful for a prediction of the pedagogical efficiency factors.

The results obtained with method 2 have been satisfying and it have been possible to determine the values of the pedagogical efficiency factors for the selected gymnasium: the educational added value for the arts exam (EAV_A) and for the science/nature exam (EAV_{SN}) at 2010 year. They have get the following values:

$$EAV_A = -0,11 \pm 0,15 \quad and \quad EAV_{SN} = +0,44 \pm 0,06$$

The forecasts for students, classes and whole gymnasium will be compared to the real exams results that will be known on June, 2010. During the research it have been concluded that average of students marks (grades) at the end of a primary school have not influence on the results (that have been expected after the talks with some gymnasium teachers). The results for the students are presented in Table 1.

Table 1. The results of a grades predication for the gymnasium students (at a stanine scale representation)

u_1	y_1	y_2
1	1,2	1,2,3
2	3,4	2,3,4
3	3,4	3,4,5
4	4,5	4,5
5	4,5,6	5,6
6	5,6	5,6,7
7	5,6,7	6,7,8
8	6,7	7,8,9
9	6,7,8	8,9

The predictions with methods 3 and 4 are carrying in the present time. The artificial neural network with one hidden layer and sigmoidal activation function have been used at first. The set of the triples: the stanin value of the competition test, an average of grades at the end of primary school and the primary school number, have been used as the input data (446 data records). The results of the art exam and of the science/nature one (both did in the past) have been the output data pairs (446 pairs). The sets of input and output data have been used in a supervised learning based on the mean-squared error method with backpropagation. In that case the network learning process have been divergent. An using of only one hidden layer was probably the cause of this divergence.

The research are continuing with an artificial neural network with three hidden layer and all other parameters as previous, without an average of grades at the end of primary school. The results are very promising: a learning process seems to be convergent.

References

1. Przybyszewski, K.: Application of the Fuzzy Sets for Evaluation of the Different Aspects of the Learning Systems. In: Automatyka, vol. 3 (12), pp. 1033–1045. AGH Academic Press, Krakow (2008) (in Polish)
2. Przybyszewski, K., Cader, A., Filutowicz, Z.: An Automation and Objectivization of the evaluation process of the learning/teaching processes. In: Cader, A., et al. (eds.) Selected Problems of the Knowledge Engineering, pp. 36–76. SWSPiZ Academic Press, Lodz (2008)
3. Niewiadomski, A., Kryger, P., Szczepaniak, P.S.: Fuzzyfication of Indiscernibility Relation for Structurizing Lists of Synonyms and Stop-Lists for Search Engines. In: Rutkowski, L., Siekmann, J.H., Tadeusiewicz, R., Zadeh, L.A. (eds.) ICAISC 2004. LNCS (LNAI), vol. 3070, pp. 504–509. Springer, Heidelberg (2004)
4. Pawlak, Z.: Rough Set Elements. In: Polkowski, L., Skowron, A. (eds.) Rough Sets in Knowledge Discovery. Springer, Heidelberg (1998)
5. Rutkowski, L.: Methods and techniques of artificial intelligence. PWN Press, Warszawa (2005) (in Polish)

6. Krus, L.: Problems of the Decision Support Systems Construction. In: Kulikowski, R., et al. (eds.) Computer and System Support for the Knowledge Management, pp. 141–151. AOW Exit, Warszawa (2006) (in Polish)
7. Przybyszewski, K., Filutowicz, Z., Cader, A.: Standardization of the Students Evaluation Results for a Prediction of the Classical and E-education System Efficiency. In: Niedwiedziski, M., Lange-Sadziska, K. (eds.) Selected Problems of the E-platforms Applifications, pp. 21–32. Consulting Press, Lodz (2009) (in Polish)
8. Tadeusiewicz, R.: Cybernetic Model of the Computer Supported Learning. In: Automatyka, vol. 3 (8), pp. 643–664. AGH Academic Press, Krakow (2004) (in Polish)
9. Przybyszewski, K.: A new evaluation method for E-learning systems. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 1209–1216. Springer, Heidelberg (2006)
10. Przybyszewski, K.: The Application of Selected AI Methods to the Evaluation of Students Progress. In: Niedwiedziski, M., Lange-Sadziska, K. (eds.) Selected Problems of E-economy, pp. 163–173. Consulting Press, Lodz (2008) (in Polish)
11. Przybyszewski, K., Cader, A., Filutowicz, Z.: Information management in the interactive e-learning systems. Scientific Biull. WSHE 4(9), 90–102 (2000) (in Polish)
12. Pokropek, A.: Accuracy of the Educational Added Value Method. In: Educationa Added Value. Part 2, CKE Research Biull. no 14, Warszawa, pp. 100–139 (2007) (in Polish)
13. Przybyszewski, K., Cader, A., Marchlewska, A.: On the Possibility of the Selected Methods of the Classical and E-learnings Systems Efficiency Methods. In: Automatyka, vol. 13(3). AGH Academic Press, Krakow (2009) (in Polish)
14. Guilford, J.P.: Fundamental Statistics in Psychology and Education. McGraw-Hill Book Company, New York (1942)

Online Speed Profile Generation for Industrial Machine Tool Based on Neuro-fuzzy Approach

Leszek Rutkowski[1,2], Andrzej Przybył[2], Krzysztof Cpałka[1,2], and Meng Joo Er[3]

[1] Academy of Management (SWSPiZ), Institute of Information Technology, Poland
[2] Czestochowa University of Technology,
Department of Computer Engineering, Poland
[3] Nanyang Technological University,
School of Electrical and Electronic Engineering, Singapore
{Leszek.Rutkowski,Andrzej.Przybyl,Krzysztof.Cpalka}@kik.pcz.pl,
emjer@ntu.edu.sg

Abstract. The paper presents the online smooth speed profile generator used in trajectory interpolation in milling machines. Smooth kinematics profile is obtained by imposing limit on the jerk - which is the first derivative of acceleration. This generator is based on the neuro-fuzzy look-ahead function and is able to adapt online the actual feedrate to changing external conditions. Such an approach improves the machining quality, reduces the tools wear and shortens total machining time.

1 Introduction

In the Computer Numerical Controlled (CNC) the desired feedrate of the machine tool not always can be achieved, because of the mechanical and electrical limitation of the machine. For example every electrical motor has limited output power, so it can produce limited component of an acceleration vector along the toolpath. The result is a limited attainable feedrate along the toolpath, suitable to it curvature (Fig. 1). Moreover the feedrate and acceleration cannot be changed abruptly, because of the possibility of exciting the natural modes of the mechanical structure or servo control system. Non smooth trajectory results in a fast wear of a mechanical components of the machine. Control system of a CNC machine should control the servo drives in such a way, that the feedrate is as close as possible to the demanded value, but the defined speed limits are not violated. Moreover, the generated trajectory should be smooth, what can be obtained by imposing limits on the first and second time derivatives of the feedrate, resulting in a trapezoidal acceleration profile. On the other hand, the trapezoidal speed profile is very popular and widely used because of its simplicity. Unfortunately, in this case there are discontinuities in acceleration reference values and they result in a low quality of the machining. The abrupt changes of the acceleration are not possible to obtain in the servo drives and it results in a substantial position tracking error. In order to avoid this, the feedrate must be limited and it results in extending total machining time. In another case, if the smooth speed profile is used, the acceleration profile has no discontinuity and it trapezoidal form results from jerk limit.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 645–650, 2010.
© Springer-Verlag Berlin Heidelberg 2010

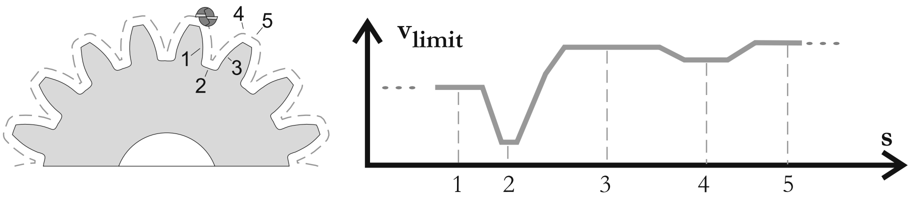


Fig. 1. Example of the geometrical path designed in the CAM system and limitation imposed on a maximum allowable velocity along it (feedrate)

The online speed profile generation methods are widely investigated in literature [1]-[2], [6]-[8]. Unfortunately none of the reported methods was able to adjust online the generated speed profile to the changing external conditions. However it should be emphasized that feedrate adaptation mechanisms, used in high precision machines, require such a feature.

2 A New Method for Online Trajectory Generation

In this work we describe a new neuro-fuzzy based method for online jerk-limited trajectory generation. This method is based on the so-called testing trajectories (Fig. 2) which are generated in defined time periods T_{TG}. The interpolation is always based on a basis of an initially known safe trajectory (thin black curve). That trajectory guides the CNC machine, along the geometrical toolpath, to the stop, guarantying the velocity, acceleration and jerk limitation. On the basis of a predicted state of the interpolator, after defined time T_{TG}, is generated one testing trajectory. The task of testing trajectory is to speed up the move, taking into account the acceleration and jerk limit values. The speed up is done by feeding a limited positive jerk value in a time interval T_{TG}. Required calculations can be realized analytically. Generated testing trajectory has to be validated, to determine if it violates or not the allowable velocity limit along the toolpath (thick black curve in Fig. 2). This velocity limit is the lower value of the technological demanded feedrate and allowable taking into account the machine dynamics (Fig. 1). If a validation algorithm classifies the testing trajectory as it violates the speed limit, this trajectory will be discarded. In other case this trajectory will be the new safe trajectory, valid after T_{TG} time period. After defined time consecutive testing trajectory will be generated, starting from the new working point. In proposed system, the testing trajectory validation algorithm plays a key role. Analytical solution of this task is very complicated, because the velocity constraints are a function of a distance while generated speed profile is a function of time. There is no simple analytical projection of the trajectory from a function of a time to a function of a distance. The trajectory validation process performed in an analytical way is very complex. Because of the mentioned difficulties with implementation of the complex calculations, we propose soft computing methods to build an efficient validation system. The neuro-fuzzy structure (NFS) aids an efficient quadratic approximation of the speed as a function of distance. Such an approximated function reduces complex calculation to the simple task

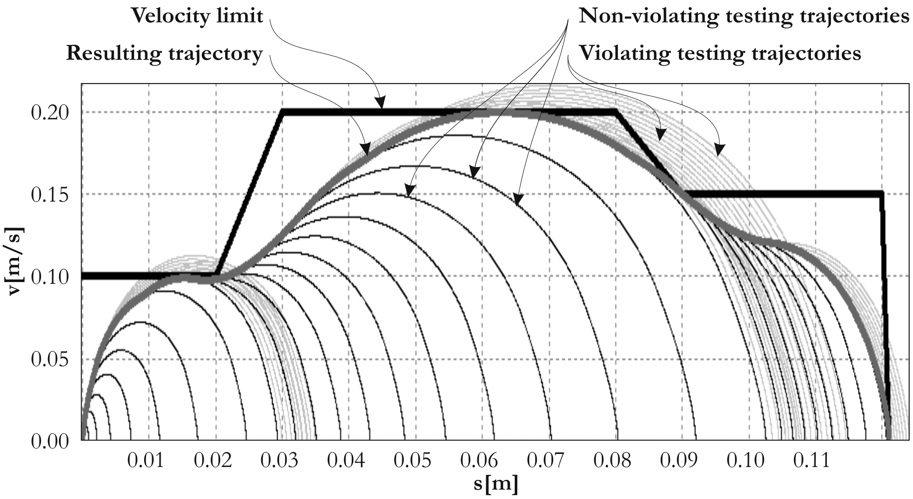


Fig. 2. Method of the online generation of the jerk limited trajectory (thick gray curve) taking into account the feedrate limitation (thick black curve). Thin gray and black curves represent test trajectories, violating and not violating velocity limitation, respectively, along corresponding distance.

of quadratic equation solving. The quadratic approximation of any continuous function is always possible with a defined acceptable error, if the approximation distance does not exceed some limited value. In our case this value depends on three parameters and can be obtained with the help of a simulation. If we know the allowable distance, we can easily and precisely approximate the function doing few simple analytical calculations. Based on such assumptions we utilize the NFS to approximate the maximum allowable distance at which the quadratic approximation accuracy is satisfactory. To approximate the whole trajectory we must repeat the quadratic approximation for all successive fragments of trajectory. Because in the trapezoidal acceleration profile the jerk can have only three different discrete values, we can use three simple NFS for three separate cases instead of a complex one. Disadvantage of this approach is the necessity to declare the used jerk values at the stage of designing control system. As a result there is no possibility to modify these values because the system is working, so any jerk adaptation algorithms are not possible to implement. However a great advantage of our approach is the simplification of the NFS and, consequently, the whole algorithm is much more efficient in real time implementation.

3 Flexible Takagi-Sugeno Neuro-fuzzy System for the Validation of the Trajectory

For the validation of the trajectory we used flexible Takagi-Sugeno neuro-fuzzy system (see e.g. [5]). Two-input, single-output system mapping $\mathbf{X} \rightarrow \mathbf{Y}$, where $\mathbf{X} \subset \mathbf{R}^2$ and $\mathbf{Y} \subset \mathbf{R}$. Assume the following the rule base

$$R^{(r)}:\left\{\begin{array}{c}\text{IF } \bar{x}_1 \text{ is } A_1^r\left(w_{1,r}^{\tau}\right) \text{ AND } \bar{x}_2 \text{ is } A_2^r\left(w_{2,r}^{\tau}\right)\\ \text{THEN } f^{(r)}\left(\bar{\mathbf{x}}\right)=c_{0,r}^{f}+\sum\limits_{i=1}^{2}c_{i,r}^{f}\bar{x}_i\end{array}\right\}\left(w_r^{def}\right), \tag{1}$$

where $r=1,2,\ldots,N$. The construction of the system is based on the following parameters and weights:

- parameters of membership functions $\mu_{A_i^k}\left(\bar{x}_i\right)$, $i=1,2$, $r=1,2,\ldots,N$,
- parameters $c_{0,r}^{f}$, $c_{i,r}^{f}$, $i=1,2$, $r=1,2,\ldots,N$, in linear models describing consequences,
- certainty weights $w_{i,r}^{\tau}\in[0,1]$, $i=1,2$, $r=1,2,\ldots,N$, describing importance of antecedents in the rules,
- certainty weights $w_r^{def}\in R$, $r=1,2,\ldots,N$, describing importance of the rules.

The aggregation in the Takagi-Sugeno model, described by the rule base (1), is in the form

$$\bar{y}=f\left(\bar{\mathbf{x}}\right)=\frac{\sum\limits_{r=1}^{N}w_r^{def}\cdot f^{(r)}\left(\bar{\mathbf{x}}\right)\cdot\mu_{A^r}\left(\bar{\mathbf{x}}\right)}{\sum\limits_{r=1}^{N}\mu_{A^r}\left(\bar{\mathbf{x}}\right)}, \tag{2}$$

where

$$\mu_{A^r}\left(\bar{\mathbf{x}}\right)=T^*\left\{\mu_{A_1^r}\left(\bar{x}_1\right),\mu_{A_2^r}\left(\bar{x}_2\right);w_{1,r}^{\tau},w_{2,r}^{\tau}\right\} \tag{3}$$

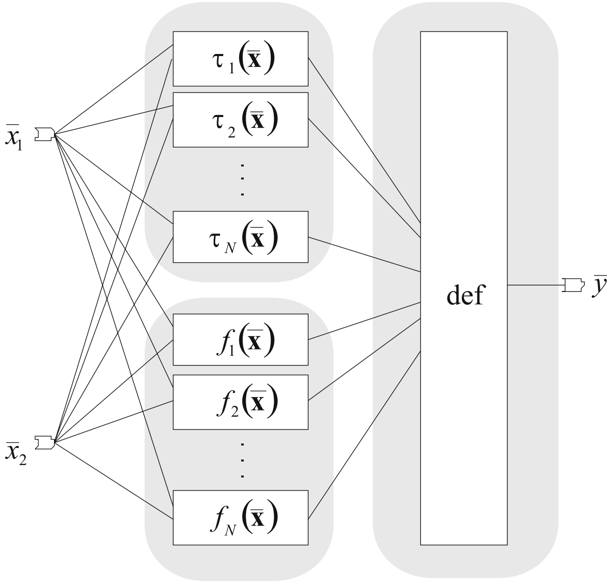


Fig. 3. Neuro-fuzzy system used for the validation of the trajectory

and T^* is a weighted t-norm (see e.g. [3], [4]). Weighted t-norm in the two-dimensional case is defined as follows

$$T^* \left\{ \mu_{A_1^r}(\bar{x}_1), \mu_{A_2^r}(\bar{x}_2); w_{1,r}^{\tau}, w_{2,r}^{\tau} \right\} = T \left\{ \begin{array}{c} 1 + \left(\mu_{A_1^r}(\bar{x}_1) - 1 \right) \cdot w_{1,r}^{\tau}, \\ 1 + \left(\mu_{A_2^r}(\bar{x}_2) - 1 \right) \cdot w_{2,r}^{\tau} \end{array} \right\}. \quad (4)$$

The weights $w_{1,r}^{\tau}$ and $w_{2,r}^{\tau}$ are certainties (credibilities) of both antecedents in (4). Observe that:

- If $w_{1,r}^{\tau} = w_{2,r}^{\tau} = 1$ then the weighted t-norm is reduced to the standard t-norm.
- If $w_{1,r}^{\tau} = 0$ then $T^* \left\{ \mu_{A_1^r}(\bar{x}_1), \mu_{A_2^r}(\bar{x}_2); 0, w_{2,r}^{\tau} \right\} = 1 + \left(\mu_{A_2^r}(\bar{x}_2) - 1 \right) \cdot w_{2,r}^{\tau}$.

The general architecture of the flexible Takagi-Sugeno system is depicted in Fig. 3. As we can see, it is a multilayer network structure. To train it, the idea of the error backpropagation method may be applied (see e.g. [3], [4]). Based on the learning sequence we determine all parameters and weights of fuzzy system (2).

4 Experimental Results

The simulations of the proposed algorithm were made on a dedicated software and the results are very satisfactory. The trajectory obtained with the help of NFS given by (2) is almost identical to the trajectory obtained based on the time consuming iterative simulations. The insignificant differences result from the NFS approximation errors.

We used neuro-fuzzy system given by (2) characterized by the Gaussian fuzzy sets, 30 rules ($N = 30$) and algebraic t-norms. We employed the FCM algorithm to find initial values of membership functions parameters ($m = 2.0$, 1000 steps). We also initialized weights of antecedents $w_{i,r}^{\tau} = 1$, $i = 1, 2$, $r = 1, 2 \ldots, N$, and weights of the rules $w_r^{agr} = 1$, $r = 1, 2 \ldots, N$. The system was learned by the backpropagation method ($\mu = 0.15$) with momentum ($\lambda = 0.10$) by 100000 epochs. The final root mean square error (RMSE) was equal 0.0996. Our approach allows to easily implement the presented algorithm in a microprocessor system used in the CNC machine.

5 Summary

In this paper we presented a new algorithm for the online speed profile generation for industrial machine tool. Our method, based on the neuro-fuzzy approach, allows the system to work properly and quickly, and to construct the trajectory generator, which can operate on-line. Moreover, it is possible to efficiently implement it in a microprocessor system.

Acknowledgment

This work was partly supported by the Polish Ministry of Science and Higher Education (Habilitation Project 2007-2010, Polish-Singapore Research Project 2008-2010, Research and Development Project 2007-2010, Research Project 2008-2011) and the Foundation for Polish Science (TEAM project 2010-2014).

References

1. Dieulot, J.-Y., Thimoumi, I., Colas, F., Béarée, R.: Numerical Aspects and Performances of Trajectory Planning Methods of Flexible Axes. International Journal of Computers, Communications & Control I(4), 35–44 (2006)
2. Osario-Rios, R.A., Romero-Trosncoso, R.d.J., Herera-Ruiz, G., Casteneda-Miranda, R.: FPGA implementation of higher degree polynomial acceleration profiles for peak jerk reduction in servomotors. Robotics and Computer-Integrated Manufacturing 25, 379–392 (2009)
3. Rutkowski, L., Cpałka, K.: Designing and learning of adjustable quasi triangular norms with applications to neuro-fuzzy systems. IEEE Trans. Fuzzy Syst. 13(1), 140–151 (2005)
4. Rutkowski, L., Cpałka, K.: Flexible neuro-fuzzy systems. IEEE Trans. Neural Networks 14(3), 554–574 (2003)
5. Takagi, T., Sugeno, M.: Fuzzy identification of systems and its application to modeling and control. IEEE Trans. Syst., Man Cybern. 15, 116–132 (1985)
6. Tsai, M.-S., Nien, H.-W., Yau, H.-T.: Development of an integrated look-ahead dynamics-based NURBS interpolator for high precision machinery. Computer-Aided Design 40, 554–566 (2008)
7. Yau, H.-T., Wang, J.-B.: Fast Bezier interpolator with real-time lookahead function for high-accuracy machining. International Journal of Machine Tools & Manufacture 47, 1518–1529 (2007)
8. Yau, H.-T., Wang, J.-B., Hsu, C.-Y., Yeh, C.-H.: PC-based Controller with Real-time Look-ahead NURBS Interpolator. Computer-Aided Design & Applications 4(1-4), 331–340 (2007)

The Design of an Active Seismic Control System for a Building Using the Particle Swarm Optimization

Adam Schmidt[1] and Roman Lewandowski[2]

[1] Poznan University of Technology, Institute of Control and Information Engineering
Adam.Schmidt@put.poznan.pl
[2] Poznan University of Technology, Institute of Structural Engineering
Roman.Lewandowski@put.poznan.pl

Abstract. Recently significant attention has been paid to the active reduction of vibrations in civil constructions. In this paper we present the synthesis of an active control system using the particle swarm optimization method. The controller design is analyzed as a building stories' displacement minimalization problem. The proposed fitness function is computationally efficient and incorporates the constraints on the system's stability and actuators' maximum output. The performance of the obtained controller was tested using historical earthquake records. The performed numerical simulations proved that the designed controller is capable of efficient vibrations reduction.

Keywords: active vibration reduction, particle swarm optimization, earthquake engineering.

1 Introduction

Recently it is common to design and construct lightweight and cost-efficient buildings. However, these light constructions are often more susceptible to vibrations caused either by human or by natural sources such as earthquakes. Therefore, it is important to design methods and technologies which would provide means to reduce the unwanted vibrations in buildings.

The active control system concept, which was proposed for the first time in a context of earthquake engineering by Yaoby Yao [1], is one of the possible solutions to that problem. It consists of a controller, sensors measuring the state of the building (displacements, velocities and accelerations) and actuators generating forces opposing to the forces induced by the environment. As the active control system introduces additional forces to the structure it is neccessary to make sure that it would not destabilize the building.

Various methods have been used to design the controller for the active control systems. These include linear quadratic regulators (LQR) [4][5], instantaneuos optimal control [6][7], linear quadratic Gaussian (LQG) [8][9][10], H_2 and H_∞ [8][11][12][13], pole placement[14], modal control[15], sliding mode control[16], fuzzy control[17] or artificial neural networks[18][19].

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 651–658, 2010.
© Springer-Verlag Berlin Heidelberg 2010

Over the last years the Particle Swarm Optimization has been succesfully used in the controller design. In [20] and [21] the PSO has been used in the optimization of the PI and PID controllers. Wang et al.[22] have applied the PSO to find the control system poles resulting in a robust control system. In [23] the PSO has been succesfully used to find the optimal feedback gain in the vehicle navigation system controller.

In this paper we present the design of the building active control system using the PSO. The constrained controller design is formulated as an optimization problem. The proposed fitness function minimizes the building's structure displacements and incorporates constraints on the system's stability and requirements concerning maximum forces generated by actuators. The effectiveness of the presented method is assessed on the model of a six-story building under different earthquakes.

2 Structure Model

The building is modeled as a shear planar frame with actuators installed between some of its stories. It is assumed that the braces supporting actuators are infinitely stiff. The motion equations of the system can be defined as:

$$M\ddot{q}(t) + C\dot{q}(t) + Kq(t) = Eu(t) + f(t) \tag{1}$$

where M, C, K and E stand for the mass, damping, stiffness and location matrices. The $q(t)$ is a vector of the stories displacements relative to the ground, $u(t)$ is a vector of the forces generated by the actuators and the $f(t)$ is a vector of the forces induced by the earthquake.

Under the assumption that the mass of each floor is lumped the mass matrix M of N-stories building takes a form of a diagonal matrix with the masses of the subsequent floors on the main diagonal:

$$M = diag(m_1, m_2, \dots m_N) \tag{2}$$

The stiffness matrix K is defined as:

$$K = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 & \dots & 0 & 0 \\ -k_2 & k_2 + k_3 & -k_3 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & -k_{N-1} & k_{N-1} + k_N & -k_N \\ 0 & 0 & \dots & 0 & -k_N & k_N \end{bmatrix} \tag{3}$$

where k_i is the stiffness of the i-th story. In our study we assumed that $N = 6$, $m_i = 10000kg$ and $k_i = 2250000\frac{N}{m}$. The damping matrix is calculated as a linear combination of the mass and stiffness matrices:

$$C = \alpha M + \beta K \tag{4}$$

$$\alpha = \frac{2\omega_1\omega_2(\gamma_1\omega_2 - \gamma_2\omega_1)}{(\omega_2^2 - \omega_1^2)} \tag{5}$$

$$\beta = \frac{2(\gamma_1\omega_2 - \gamma_2\omega_1)}{(\omega_2^2 - \omega_1^2)} \tag{6}$$

where $\omega_1 = 3.6161\frac{rad}{s}$ and $\omega_2 = 10.6381\frac{rad}{s}$ are the structural modal frequencies of the first and the second mode of the uncontrolled system and $\gamma_1 = 0.02$ and $\gamma_2 = 0.02$ are the dimensionless modal damping ratios.

The location matrix E defines the actuators positions in the building structure. We considered a building with two actuators placed below the first and the third stories. The resulting location matrix took form:

$$E = \begin{bmatrix} -1 & 0 \\ 0 & 1 \\ 0 & -1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{7}$$

The vector of earthquake induced forces is calculated as:

$$f(t) = M \cdot 1_{N\times 1} a_{gr}(t) \tag{8}$$

where $a_{gr}(t)$ is the acceleration of the ground.

The forces generated by the actuators are calculated according to the structure displacements $q(t)$ and velocities $\dot{q}(t)$. It is assumed, that all displacements and velocities are measured, which means that:

$$u(t) = -G_1 q(t) - G_2 \dot{q}(t) \tag{9}$$

where G_1 and G_2 are the gain matrices of the control system feedback loop. The Eqn. 1 can be rewritten as:

$$M\ddot{q}(t) + (C + EG_2)\dot{q}(t) + (K + EG_1)q(t) = M \cdot 1_{N\times 1} a_{gr}(t) \tag{10}$$

The model can be presented in the following form of the state equations:

$$z(t) = \begin{bmatrix} q(t) \; \dot{q}(t) \end{bmatrix}^T \tag{11}$$

$$\dot{z}(t) = Az(t) + Ba_{gr}(t) \tag{12}$$

$$A = \begin{bmatrix} 0_{(N\times N)} & 1_{(N\times N)} \\ -M^{-1}(K + EG_1) & -M^{-1}(C + EG_2) \end{bmatrix} \tag{13}$$

$$B = \begin{bmatrix} 0_{1\times N} \; 1_{1\times N} \end{bmatrix}^T \tag{14}$$

3 Particle Swarm Optimization

3.1 Description

The Particle Swarm Optimization (PSO) introduced by Kennedy and Eberhart [2] is an population-based optimization technique inspired by social behaviour of animals e.g. birds flocking or fish schooling. Similarly to the genetic algorithm the populations consists of possible solutions (called particles) and the search for an optimal solutions is performed by updating subsequent generations. However, no evolutionary operators such as cross-over or mutation are used. Instead, each particle explores the problem space being drawn to the current optimal solutions.

The main difference between GA and PSO is the memory of particles. Each particle keeps a record of its best fitness achieved so far (along with the associated solution Pb) and the best fitness and corresponding solution achieved in the particle's neighborhood - Lb. It was shown that using the global neighborhood (all particles are fully aware of other particles' fitness) minimizes the median number of iterations needed to converge. On the other hand the neighborhood of size 2 gives the highest resistance to local minima.

At each iteration i of the PSO the velocities of the particles are changed (accelerated) towards the Pb and the Lb and the particles are moved to new positions:

$$v_j(i) = w \cdot v_j(i-1) + c_1 \cdot rand \cdot (Pb_j - p_j(i)) + c_2 \cdot rand \cdot (Lb_j - p_j(i)) \quad (15)$$

$$p_j(i) = p_j(i-1) + v_j(i) \quad (16)$$

where v_j and p_j are the velocity and the position of the j-th particle, w is the intertia factor providing balance between the exploration and the exploitation, c_1 is the individuality constant and c_2 is the sociality constant.

To avoid the "velocity explosion" the maximum velocity constraint $v_{max} = 100000$ is introduced. Whenever the velocity violates the $[-v_{max}, v_{max}]$ limits it is truncated to that range. Additionally, to speed up the convergence the inertia weight was linearly reduced from $w_{max} = 0.9$ to $w_{min} = 0.1$.

In our experiments we have used $n = 20$ particles, the maximum number of iterations $i_{max} = 200000$ and $c_1 = c_2 = 2$. The neighborhood of size 4 was used as a tradeoff between the fast convergence and the resistance to local minima. More information on the parameters selection of the algorithm and its variations can be found in [3].

3.2 Fitness Function and Covergence Criterion

The optimization goal was to find the gain matrices G_1 and G_2 that would minimize the displacements of building's stories under the earthquake. Additionally, the resulting model had to be stable and the generated forces had to be lower than an assumed value ($F_{max} = 100000N$). Those constraints were incorporated into the fitness function:

$$\text{fit}(p) = \text{fit}_{displacement}(p) + a \cdot \text{fit}_{stability}(p) + b \cdot \text{fit}_{forces}(p) \quad (17)$$

where a and b are constraints coefficients.

The displacements of the structure were analyzed in the frequency domain under the simplifying assumption that the ground acceleration is a sinusoidal signal. The following transfer function was defined:

$$H_{disp}(j\omega) = \frac{Q(j\omega)}{A_{gr}(j\omega)} = (-M\omega^2 + j(C + E \cdot G_2)\omega + K + E \cdot G_1)^{-1} M \cdot 1_{N\times 1} \quad (18)$$

where $Q(j\omega)$ and A_{gr} are the Fourier transforms of the displacements and ground acceleration respectively.

The biggest (and thus the most dangerous for the structure) displacements are generated for the modal frequencies of the resulting system. Therefore, the following fitness function component was defined:

$$\mathrm{fit}_{displacement}(p) = \sum_{i=1}^{N} 1_{1\times N} |H_{disp}(j\omega_i)| \tag{19}$$

where ω_i is the i-th modal frequency of the closed-loop system.

The resulting system would be stable if the real parts of all the system's poles were smaller than 0. The fitness function stability component was calculated as:

$$\mathrm{fit}_{stability}(p) = \begin{cases} 1 + \max(\Re(e_i)) - \rho & \text{if } \max(\Re(e_i)) \geq \rho \\ 0 & \text{if } \max(\Re(e_i)) < \rho \end{cases} \tag{20}$$

where e_i is the i-th eigenvalue of the state matrix A and ρ is the maximal allowed real part of the system's poles.

It was assumed that the actuators should not saturate until the ground acceleration amplitude reached a certain value ($A_{max} = 1\frac{m}{s^2}$) at any of the system's modal frequencies. The following transfer function was defined:

$$H_{force}(j\omega) = \frac{U(j\omega)}{A_{gr}(j\omega)} = (-G_1 - jG_2\omega)H_{disp}(j\omega) \tag{21}$$

The force component of the fitness function was calculated according to:

$$\mathrm{fit}_{force}(p) = \begin{cases} \max\left(\frac{|H_{force}(j\omega)|A_{max}}{F_{max}}\right) & \text{if } \max\left(\frac{|H_{force}(j\omega)|A_{max}}{F_{max}}\right) > 1 \\ 0 & \text{if } \max\left(\frac{|H_{force}(j\omega)|A_{max}}{F_{max}}\right) \leq 1 \end{cases} \tag{22}$$

The a parameter was set to 1000000 and the b was set to 1000. This ensured that any solution resulting in an unstable system would have higher fitness function value than any of the stable ones and that solutions violating the maximum force limits would have worse fitness than those conforming to both constraints.

The proposed fitness function can be calculated without the time consuming simulations which is an important advantage in any iterative optimization algorithm.

The convergence of the PSO was assumed if the best fitness value in the population had not changed over $i_{conv} = 5000$ iterations and the best fitness value had been smaller than $\min(a, b)$ meaning that the solution conformed to both constraints.

4 Results

The optimization process was executed 100 times. On average the algorithm needed 20848 iteration to converge (median = 19856). The best result obtained

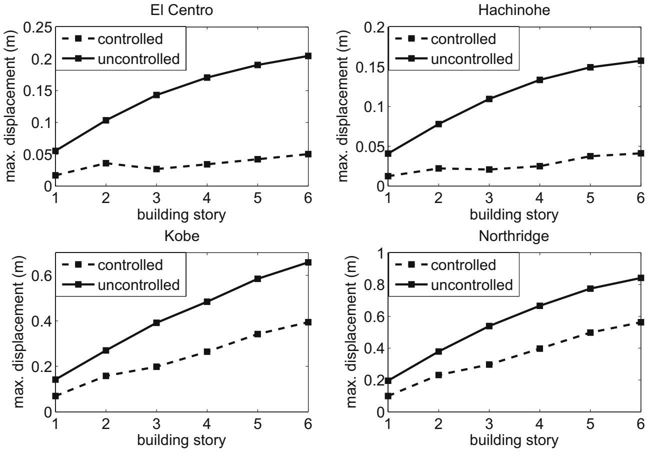


Fig. 1. The maximum displacement of building stories

Table 1. The normalized RMS of building displacement

El Centro	Hachinohe	Kobe	Northridge
0.1525	0.1703	0.353	0.4469

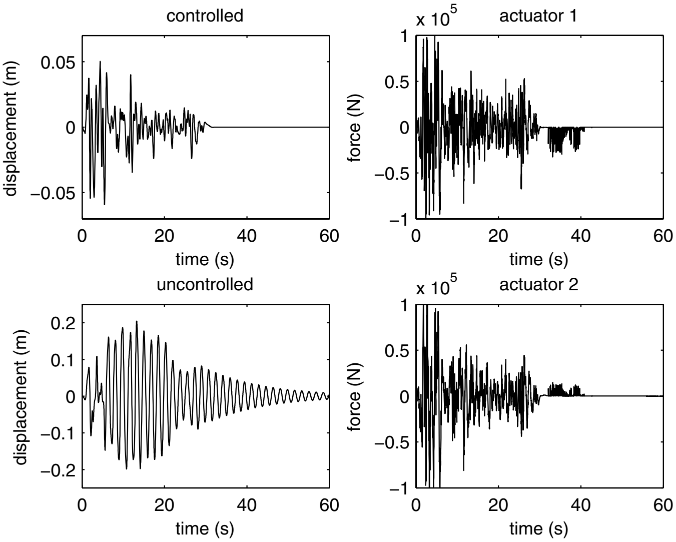


Fig. 2. The displacement of the top floor(left) and the forces generated by the actuators (right) under the El Centro earthquake

was equal to 0.1745, average fitness function of the converged solutions was equal to 0.3784 with the standard deviation of 0.0825 .

The performance of the obtained controller was tested in numerical simulations. Four different earthquake records were used: El Centro, Hachinohe, Kobe, Northridge. The peak ground accelerations of these earthquakes were: $0.3188g$, $0.2294g$, $0.8337g$, $0.8428g$ respectively. The maximum displacement of the building stories is shown in the Figure 1. The example of the top story displacement of both controlled and uncontrolled building as well as the forces generated by the actuators are shown in the Figure 2. Additionally, the normed RMS of structure displacement was calculated for all of the considered earthquakes (Table 1):

$$D_{RMS} = \frac{\max_i \sqrt{\frac{1}{T}\int_0^T q_i^c(t)^2 \mathrm{dt}}}{\max_i \sqrt{\frac{1}{T}\int_0^T q_i^{uc}(t)^2 \mathrm{dt}}} \tag{23}$$

where i is the story number, T is the duration time of the earthquake, q_i^c is the displacement of the i-th story of the controlled building and q_i^{uc} is the displacement of the i-th story of the uncontrolled building.

5 Conclusions

This paper presents a successfull attempt to design an active vibrations control system using the PSO technique. To the extent of the authors' knowledge this was the first attempt to design an active building controller by optimization of static, direct feedback gain matrices. A novel, computationally efficient fitness function minimizing stories displacements and incorporating the control system constraints was defined.

The obtained controller was tested under different historical earthquake loads. It achieved excellent results in the terms of the absolute stories displacements reduction and the normalized RMS of displacements.

The presented study shows that the PSO can be succesfully used to design active control systems. In our future work we will focus on modifying the fitness function to take into account the accelerations of building stories as well as on incorporating the number and positions of sensors and actuators into the optimization scheme. Moreover, the performance of different optimization methods such as simulated annealing or evolutionary algorithms will be tested.

References

1. Yao, J.T.P.: Concept of structural control. ASCE J. Struct. Div. 98, 1567–1575 (1972)
2. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995)
3. Clerc, M., Kennedy, J.: The particle swarm - explosion, stability and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 6(1), 58–73 (2002)

4. Yamada, K., Kobori, T.: Linear quadratic regulator for structure under on-line predicted future seismic excitation. Earthq. Eng., Struct. Dyn. 25, 631–644 (1996)
5. Ikeda, Y.: Effect of weighting a stroke of an active mass damper in the linear quadratic regulator problem. Earthq. Eng., Struct. Dyn. 26, 1125–1136 (1997)
6. Chang, C.C., Yang, H.T.Y.: Instantaneous optimal control of building frames. J. Struct. Eng. 120, 1307–1326 (1994)
7. Wong, K.K.F., Yang, R.: Predictive instantaneous optimal control of elastic structures during earthquakes. Earthq. Eng., Struct. Dyn. 32, 2161–2177 (2003)
8. Wu, J.C., Yang, J.N., Schmitendorf, W.E.: Reduced-order H_∞ and LQR control for wind-excited tall buildings. Eng. Struct. 20, 222–236 (1998)
9. Pham, K.D., Jin, G., Sain, M.K., Spencer, B.F., Liberty, S.R.: Generalized linear quadratic Gaussian technique for wind benchmark problem. J. Eng. Mech. 130, 466–470 (2004)
10. Xu, Y.L., Zhang, W.S.: Closed-form solution for seismic response of adjacent buildings with linear quadratic Gaussian controllers. Earthq. Eng., Struct. Dyn. 31, 235–259 (2002)
11. Ankiredi, S., Yang, H.T.T.: Sampled-data H_2 optimal output feedback control for civil structures. Earthq. Eng., Struct. Dyn. 28, 921–940 (1999)
12. Kose, I.E., Schmitendorf, W.E., Jabbari, F., Yang, J.N.: H_∞ active seismic response control using static output feedback. J. Eng. Mech. 122, 651–659 (1996)
13. Jabbari, F., Schmitendorf, W.E., Yang, J.N.: H_∞ control for seismic-excited buildings with acceleration feedback. J. Eng. Mech. 121, 994–1002 (1995)
14. Chang, C.C., Yu, L.O.: A simple optimal pole location technique for structural control. Eng. Struct. 20, 792–804 (1998)
15. Lu, I.Y.: Discrete-time modal control for seismic structures with active bracing system. J. Inteligent Sys. Struct. 12, 369–381 (2001)
16. Moon, S.J., Bergman, L.A., Voulgaris, P.G.: Sliding mode control of cable-stayed bridge subjected to seismic excitation. J. Eng. Mech. 129, 71–78 (2003)
17. Ashlawat, A.S., Ramaswamy, A.: Multiobjective optimal fuzzy logic controller driven active and hybrid control systems for seismically excited nonlinear buildings. J. Eng. Mech. 130, 416–423 (2004)
18. Bani-Hani, K., Ghaboussi, J.: Neural networks for structural control of a benchmark problem, active tendon system. Earthq. Eng.: Struct. Dyn. 27, 1225–1245 (1998)
19. Przychodzki, M., Lewandowski, R.: Optimal active control of building structures based on acceleration feedback using artificial neural networks. In: Proceedings of the Computer Methods in Mechanics (2007)
20. Qiao, W., Venayagamoorthy, G., Harley, R.: Design of optimal PI controllers for doubly fed induction generators driven by wind turbines using particle swarm optimization. In: Proceedings of 2006 International Joint Conference on Neural Networks, pp. 1982–1987 (2006)
21. Gaing, Z.-L.: A particle swarm optimization approach for optimum design of PID controller in AVR system. IEEE Transaction on Energy Conversion 19(2), 384–391 (2004)
22. Wang, J., Brackett, B., Harley, R.: Particle Swarm-Assisted State Feedback Control: From Pole Selection to State Estimation. In: 2009 American Control Conference, pp. 1493–1498 (2009)
23. Sun, T.-Y., Huang, C.-S., Tsai, S.-J.: Particle Swarm Optimizer based controller design for Vehicle Navigation System. In: IEEE International Conference on Systems, Man and Cybernetics, pp. 909–914 (2008)

The Normalization of the Dempster's Rule of Combination

Pavel Sevastjanov, Pavel Bartosiewicz, and Kamil Tkacz

Institute of Comp.& Information Sci.,Czestochowa University of Technology,
Dabrowskiego 73, 42-200 Czestochowa, Poland
sevast@icis.pcz.pl

Abstract. A new method for normalization and combination of interval-valued belief structures within the framework of Dempster-Shafer theory of evidence based on the so-called "interval extended zero" method is proposed. The two desirable intuitively obvious properties of normalization procedure are defined. The main of them is based on the assumption that the sum of normalized intervals should be an interval centered around 1 with a minimal width. The advantages of a new method are illustrated with use of numerical examples. It is shown that a new method performs better than known methods for combination of interval-valued belief structures as it provides the results with the properties which are close to the desirable ones.

Keywords: Interval Dempster's rule of combinaion, Normalization.

1 Introduction

Dempster-Shafer theory of evidence is concerned with bodies of evidence, which are assignments of weights to crisp events such as fault occurrence [6]. Given a domain X of possible events, a basic assignment m is a mapping $P(X) \rightarrow [0,1]$ where $P(X)$ is the power set (set of all subsets) of the domain. For any element $A \in P(X)$, the basic assignment $m(A)$ gives the amount of evidence that support that event and no other. Basic assignments are required to follow the axioms: $m(\emptyset) = 0, \sum\limits_{A \in P(X)} m(A) = 1$. The core of the evidence theory is the Dempsters rule of combination of evidence from different sources. The rule assumes that information sources are independent and uses the so-called orthogonal sum to combine multiple belief structures. With two belief structures m_1, m_2, the Dempster's rule of combination is defined as

$$m_{12}(A) = \frac{\sum\limits_{B \cap C = A} m_1(B)m_2(C)}{1-K}, A \neq \emptyset, m_{12}(\emptyset) = 0, \tag{1}$$

where $K = \sum\limits_{B \cap C = \emptyset} m_1(B)m_2(C)$. The resulting belief structures is normalized $(\sum\limits_{A \in P(X)} m_{12}(A) = 1)$. The problem arises when the belief structures m_1, m_2 are

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 659–666, 2010.
© Springer-Verlag Berlin Heidelberg 2010

presented by intervals. The use in of usual interval arithmetic rules in (1) leads to the resulting interval belief structures m_{12} representing by too wide intervals [2] which are substantially wider than the initial interval belief structures m_1, m_2. Therefore, in [2,8] the quadratic programming models for interval Dempster's rule of combination were proposed. These models lead to the interval results, but the question arises: are they normalized? It is well known that the sum of intervals can be equal to 1 (or any real value) if at least one of the intervals is inverted one, i.e., its right bound is greater than the left one. So the properties of interval normalization should be defined. Such properties which can be treated also as the definition of normalization are proposed in [7] as follows.

Let $N = \left\{ X = (x_1, \ldots, x_n) | \hat{w}_i^L \leq x_i \leq \hat{w}_i^U, i = 1, \ldots, n, \sum\limits_{i=1}^{n} x_i = 1 \right\}$ be a set of normalized vectors. Then interval vector $[\hat{w}]$ =($[\hat{w}_1^L, \hat{w}_1^U]$,...,$[\hat{w}_n^L, \hat{w}_n^U]$) is said to be normalized if and only if it satisfies the following two conditions:
(1). There exists at least one normalized vector $X = (x_1, \ldots, x_n)$ in the set N (N is none empty);
(2) All $\hat{w}_i^L$ and $\hat{w}_i^U$, $i = 1, \ldots, n$ are attainable in N.
Conditions (1) and (2) are rather of mathematical nature and do not form an exhaustive set of desirable properties of interval sum normalization.

The firs such property can be formulated as follows. As the sum of normalized real valued elements of interval belief structures m_{12} is always equal to 1, it seems natural to require that it should be equal to "near 1", where "near 1" may be presented by an interval with a minimal width.

The second desirable property of interval weights normalization may be the remaining of ratios between the means of normalized intervals as close as possible to those of initial intervals as the normalization of real valued weights does not change these ratios at all.

In the current report, we propose a new method for normalizing the Dempster's rule of combination which provides the results with above two properties. The rest of the paper is set out as follows. In Section 2, we recall the basics of the so called "interval extended zero" method for the solution of linear interval equations [4,5] and show that it can be used for the normalization of interval sum. Section 3 is devoted to the comparison of the results obtained with use of approaches developed in [2,8] and a new method proposed in the current report. Section 4 concludes with some remarks.

2 The Use of "Interval Extended Zero" Method for Interval Sum Normalization

As the equation $\sum\limits_{i=1}^{n} [\hat{w}_i] = 1$ is not verified for positive regular intervals $[\hat{w}_i]$, $i = 1, \ldots, n$, we propose to find such interval normalization factor $[x]$ that

$$\sum_{i=1}^{n} [\hat{w}_i][x] = \text{"}near\, 1\text{"}, \tag{2}$$

where "near 1" may be presented by an interval too. The more precise definition of "near 1" may be obtained using the following reasoning [4,5]: "Without loss of generality, we can define the degenerated (usual) zero as the result of operation a-a, where a is any real valued number or variable. Hence, in a similar way we can define an "interval zero" as the result of operation $[a]$ - $[a]$, where $[a]$ is an interval. It is easy to see that for any interval $[a]$ we get $[\underline{a},\overline{a}]$-$[\underline{a},\overline{a}]$=$[\underline{a}-\overline{a},\overline{a}-\underline{a}]$. Therefore, in any case, the result of interval subtraction $[a]$-$[a]$ is an interval centered around 0." Similarly we can define the "near 1" as an interval centered around 1. The problem (2) can be solved with use of "interval extended zero" method [4,5]. Let us denote $[a]=\sum\limits_{i=1}^{n}[\hat{w}_i]$. Then the problem $\sum\limits_{i=1}^{n}[\hat{w}_i]=1$ can be presented in the more general form:

$$[a][x]-[b]=0, \tag{3}$$

where $[a]$, $[b]$ are positive intervals or real values. Of course, when $[a]$ or $[b]$ or both are intervals, the equation (3) has no solution as in the left hand side we have interval, whereas the right hand side is the degenerated zero (non-interval value). It is shown in [4,5] that if we treat the equation (3) as the result of interval extension of usual real valued equation $ax-b=0$ then the right hand side of this equation (zero) should be extended too using above definition of "interval zero". It must be emphasized that this definition says nothing about the width of "interval zero". Really, when extending equation with previously unknown values of variables in the left hand side, only what we can say about the right hand side is that it should be interval symmetrical with respect to 0 with not defined width. Hence, as the result of interval extension of $ax-b=0$ in general case we get

$$[\underline{a},\overline{a}][\underline{x},\overline{x}]-[\underline{b},\overline{b}]=[-y,y]. \tag{4}$$

Of course, the value of y in Eq.(4) is not yet defined and this seems to be quite natural since the values of $\underline{x}$, $\overline{x}$ are also not defined. In the case of positive intervals $[a]$ and $[b]$, i.e., $\underline{a},\overline{a},\underline{b},\overline{b}>0$ from Eq. (4) we get

$$\{\,\underline{ax}-\overline{b}=-y,\overline{ax}-\underline{b}=y. \tag{5}$$

Finally, from Eq. (5) we obtain only one linear equation with two unknown variables $\underline{x}$ and $\overline{x}$:

$$\underline{ax}+\overline{ax}-\underline{b}-\overline{b}=0. \tag{6}$$

If there are some restrictions on the values of unknown variables $\underline{x}$ and $\overline{x}$, then Eq. (6) with these restrictions may be considered as the so called Constraint Satisfaction Problem (CSP)[1] and an interval solution may be obtained. The first restriction on the variables $\underline{x}$ and $\overline{x}$ is a solution of Eq. (6) assuming $\underline{x}=\overline{x}$. In this degenerated case, we get the solution of Eq. (6) as $x_m=\frac{\underline{b}+\overline{b}}{\underline{a}+\overline{a}}$. It is easy to see that x_m is the upper bound for $\underline{x}$ and the lower bound for $\overline{x}$ (if $\underline{x}>x_m$ or $\overline{x}<x_m$ we get an inverted solution of Eq. (6)). The natural low bound for $\underline{x}$ and

upper bond for $\overline{x}$ may be defined using basic definitions of interval arithmetic [3] as $\underline{x}=\frac{\underline{b}}{\overline{a}}$, $\overline{x} = \frac{\overline{b}}{\underline{a}}$.

Thus, we have $[\underline{x}] = [\frac{\underline{b}}{\overline{a}}, x_m]$ and $[\overline{x}] = [x_m, \frac{\overline{b}}{\underline{a}}]$. These intervals can be narrowed taking into account Eq. (6), which in the spirit of CSP is treated as a restriction. It is clear that the right bound of $\underline{x}$ and left bound of $\overline{x}$, i.e., x_m, can not be changed as they present the degenerated (crisp) solution of (6). So let us focus of the left bound of $\underline{x}$ and right bound of $\overline{x}$. From (6) we have

$$\underline{x} = \frac{\underline{b} + \overline{b} - \overline{a}\overline{x}}{\underline{a}}, \overline{x} \in [x_m, \frac{\overline{b}}{\underline{a}}], \overline{x} = \frac{\underline{b} + \overline{b} - \underline{a}\underline{x}}{\overline{a}}, \underline{x} \in [\frac{\underline{b}}{\overline{a}}, x_m]. \tag{7}$$

It is shown in [5] that for the positive $[a]$ and $[b]$, Eq. (7) have the following interval solution:

$$[\underline{x}] = \left[\underline{x}_{\max}, \frac{\underline{b} + \overline{b}}{\underline{a} + \overline{a}}\right], [\overline{x}] = \left[\frac{\underline{b} + \overline{b}}{\underline{a} + \overline{a}}, \overline{x}_{\min}\right], \tag{8}$$

where $\underline{x}_{\max} = \frac{\underline{b}}{\overline{a}}$, $\overline{x}_{\min} = \frac{\underline{b}+\overline{b}}{\underline{a}} - \frac{\underline{a}\underline{b}}{\overline{a}^2}$. Expressions (8) define all possible solutions of Eq. (4). The values of $\overline{x}_{min}$, $\underline{x}_{max}$ constitute an interval which produce the widest interval zero after substitution of them in Eq. (4). In other words, the maximum interval solution's width $w_{\max} = \overline{x}_{\min} - \underline{x}_{\max}$ corresponds to the maximum value of y: $y_{\max} = \frac{\overline{a}\overline{b}}{\underline{a}} - \underline{b}$. Substitution of degenerated solution $\underline{x} = \overline{x} = x_m$ in Eq. (4) produces the minimum value of y: $y_{min} = \frac{\overline{a}\cdot\overline{b} - \underline{a}\cdot\underline{b}}{\underline{a}+\overline{a}}$. Obviously, for any permissible solution $\underline{x}' > \underline{x}_{max}$ there exists corresponding $\overline{x}' < \overline{x}_{min}$, for each $\underline{x}'' > \underline{x}'$ the inequalities $\overline{x}'' < \overline{x}'$ and $y'' < y'$ take place. Thus, the formal interval solution (8) factually represents the continuous set of nested interval solutions of Eq.(6) which can be naturally interpreted as a fuzzy number [4,5]. It is seen that values of y characterize the closeness of the right hand side of Eq. (4) to the degenerated zero and the minimum value y_{min} is defined exclusively by interval parameters $[a]$ and $[b]$. Hence, the values of y may be considered, in a certain sense, as a measure of interval solution's uncertainty caused by the initial uncertainty of Eq. (4). Therefore, the following expression was introduced in [4,5]:

$$\alpha = 1 - \frac{y - y_{min}}{y_{max} - y_{min}}, \tag{9}$$

which may be treated as a certainty degree of interval solution of Eq. (4). We can see that α rises from 0 to 1 with decreasing of interval's width from maximum value to 0, i.e., with increasing of solution's certainty. Consequently, the values of α may be treated as labels of α-cuts representing some fuzzy solution of Eq. (4). Finally, the solution is obtained in [4,5] in form of triangular fuzzy number

$$\tilde{x} = \left\{\underline{x}_{\max}, \frac{\underline{b} + \overline{b}}{\underline{a} + \overline{a}}, \overline{x}_{\min}\right\}. \tag{10}$$

Obviously, we can assume the support of obtained fuzzy number to be a solution of analyzed problem. Such a solution may be treated as the "pessimistic" one since it corresponds to the lowest α-cuts of resulting fuzzy value. The word "pessimistic"is used here to emphasize that this solution is charged with the largest imprecision as it is obtained in the most uncertain conditions possible on the set of considered α -cuts. On the other hand, it seems natural to utilize all additional information available in the fuzzy solution. The resulting fuzzy solution can be reduced to the interval solution using well known defuzzification procedures. In the considered case, the defuzzified left and right boundaries of the solution can be represented as

$$\underline{x}_{def} = \frac{\int_0^1 \underline{x}(\alpha) d\alpha}{\int_0^1 d\alpha}, \overline{x}_{def} = \frac{\int_0^1 \overline{x}(\alpha) d\alpha}{\int_0^1 d\alpha} \tag{11}$$

In the case of $[a], [b] > 0$, from (4) and (9) the expressions for $\underline{x}(\alpha)$ and $\overline{x}(\alpha)$ can be obtained. Substituting them into (11) we get

$$\underline{x}_{def} = \frac{\overline{b}}{\underline{a}} - \frac{y_{max} + y_{min}}{2\underline{a}}, \overline{x}_{def} = \frac{\underline{b}}{\overline{a}} + \frac{y_{max} + y_{min}}{2\overline{a}}. \tag{12}$$

Since Eq.(4) is the interval extension of Eq.(3) which can be presented in algebraically equivalent form $[x] = \frac{[b]}{[a]}$, this solution can be considered also as the result of modified interval devision $[x]_{\text{mod}} = \left(\frac{[b]}{[a]}\right)_{\text{mod}}$. The modified interval devision can be treated as an alternative to the conventional interval devision: $\underline{x} = \frac{\underline{b}}{\overline{a}}$, $\overline{x} = \frac{\overline{b}}{\underline{a}}$.

It is shown in [4,5] that the proposed method provides the considerable reducing of resulting interval's length in comparison with that obtained using conventional interval arithmetic rules.

In the following we shall use the introduced modified interval devision instead of conventional interval division in the interval Dempster's rule of combination. Let us turn to the normalization problem. To obtain the interval normalization factor $[x]$ in Eq. (2), we rewrite Eq. (4) as follows

$$\sum_{i=1}^{n} [w_i] \cdot [x] - [1, 1] = [-y, y].$$

The solution is obvious and can be obtained substituting $\sum\limits_{i=1}^{n} [w_i]$ instead of $[a]$ and 1 instead of $\underline{b}$, $\overline{b}$, in (12). As the result we get

$$[x] = \left[\frac{1}{\sum\limits_{i=1}^{n} w_i^U}, \frac{2}{\sum\limits_{i=1}^{n} w_i^U} - \frac{\sum\limits_{i=1}^{n} w_i^L}{\left(\sum\limits_{i=1}^{n} w_i^U\right)^2} \right]. \tag{13}$$

Thus, the normalization procedure can be presented as

$$[\hat{w}_i] = [w_i] \cdot [x], i = 1, ..., n. \tag{14}$$

It is easy to see that for normalized $[\hat{w}_i]$, $i = 1, ..., n$ always $\sum\limits_{i=1}^{n} [\hat{w}_i] \subset [0, 2]$ and the sum $\sum\limits_{i=1}^{n} [\hat{w}_i]$ is an interval centered around 1. In our case, the expressions (14) take form:

$$[x]_{def} = \left[\frac{1}{\sum\limits_{i=1}^{n} w_i^L} - \frac{y_{\max} + y_{\min}}{2\sum\limits_{i=1}^{n} w_i^L}, \frac{1}{\sum\limits_{i=1}^{n} w_i^U} - \frac{y_{\max} + y_{\min}}{2\sum\limits_{i=1}^{n} w_i^U} \right], \qquad (15)$$

where $y_{\max} = 1 - \frac{\sum\limits_{i=1}^{n} w_i^L}{\sum\limits_{i=1}^{n} w_i^U}$, $y_{\min} = \frac{\sum\limits_{i=1}^{n} w_i^U - \sum\limits_{i=1}^{n} w_i^L}{\sum\limits_{i=1}^{n} w_i^L + \sum\limits_{i=1}^{n} w_i^U}$.
The normalization is presented as follows:

$$[\hat{w}_i] = [w_i] \cdot [x]_{def}, i = 1, ..., n. \qquad (16)$$

3 The Comparison of a New Method with Known Approaches to the Normalization of Interval Dempster's Rule of Combination

As a base of comparison, we analyze here three examples with the frame of discernment consist of three elements $H = (H_1, H_2, H_3)$ and two sources of evidence.

Example 1:
$m_1(H_1) = [0.1, 0.3], m_1(H_2) = [0.2, 0.5], m_1(H_3) = [0.1, 0.4], m_1(H) = [0, 0.4]$,
$m_2(H_1) = [0.1, 0.5], m_2(H_2) = [0.2, 0.3], m_2(H_3) = [0.1, 0.4], m_2(H) = [0.1, 0.3]$.
Example 2:
$m_1(H_1) = [0.2, 0.4], m_1(H_2) = [0.2, 0.4], m_1(H_3) = [0.1, 0.2], m_1(H) = [0.2, 0.3]$,
$m_2(H_1) = [0.2, 0.4], m_2(H_2) = [0.2, 0.4], m_2(H_3) = [0.1, 0.2], m_2(H) = [0.2, 0.3]$.
Example 3:
$m_1(H_1) = [0, 0.2], m_1(H_2) = [0.3, 0.4], m_1(H_3) = [0.2, 0.3], m_1(H) = [0.2, 0.4]$,
$m_2(H_1) = [0.2, 0.4], m_2(H_2) = [0, 0.2], m_2(H_3) = [0.1, 0.6], m_2(H) = [0.1, 0.4]$.

The numbers of the above examples correspond to the numbers in Table 1. In the framework of a new method, we use the results presented in the previous Section twice. We use the usual interval addition, multiplication and subtraction operations [3] in the Dempster's rule of combination (1) when the belief structures m_1, m_2 are presented by intervals, but to get the interval results more narrow we use in (1) the modified interval division $[x]_{\text{mod}} = \left(\frac{[b]}{[a]}\right)_{\text{mod}}$ based on the expressions (12). The resulting belief structure is not yet normalized as the sum $\sum\limits_{A \in P(X)} m_{12}(A)$ is not generally an interval centered around 1. Therefore, to get the result of the Dempster's rule of combination with two desirable properties formulated in Section 1, we use additionally the normalization procedure based on the expressions (15),(16). The results are denoted in Table 1 as New.

We compare the results we get with the use of a new method with those obtained using the methods developed by Denoeux [2] and by Wang et al. [8]. The results obtained using the Denoeux's and Wang's methods generally have no the first desirable property formulated in Section 1. Therefore, to make these results comparable with those obtained using a new method we additionally normalized them using the expressions (15),(16). The corresponding results are denoted in Table 1 as the Denoeux's and Wang's respectively.

To estimate the degree to which the second desirable property (the remaining of ratios between the means of normalized intervals as close as possible to those of initial intervals) is satisfied, the following expression has been used $\sigma = \frac{1}{n(n-1)}\sqrt{\sum\limits_{i=1}^{n}(s_i - \hat{s}_i)^2}$, where s_i are the centers of intervals before the normalization with the use expressions (15),(16), $\hat{s}_i$ are the centers of intervals after normalization. Of course, the lower the σ, the better the second desirable property is satisfied.

The results of comparison of the analised methods are presented in Table 1, where the sum $\sum\limits_{A\in P(X)} m_{12}(A)$ is denoted as $\sum m_{12}$.

Table 1. The results of comparison

N	Denoeux's		Wang's		New	
	$\sum m_{12}$	σ	$\sum m_{12}$	σ	$\sum m_{12}$	σ
1	[0.097, 1.903]	0.0282	[0.208, 1.792]	0.0179	[0.61, 1.39]	0.0252
2	[0.337, 1.663]	0.0049	[0.453, 1.547]	0.0029	[0.664, 1.336]	0.0014
3	[0.176, 1.824]	0.0296	[0.249, 1.751]	0.0081	[0.584, 1.416]	0.0074

It is easy to see that in all cases the use of the proposed normalization method (expressions (15),(16)) guarantee that for the combined evidence the sum

$$\sum_{A\in P(X)} m_{12}(A)$$

is always an interval centered around 1. Nevertheless, the resulting intervals obtained using a new method are substantionally narrower than those we get on the base of the Denoeux's and Wang's methods. So a new method satisfies the first desirable property defined in Section 1 in the more extent. We can see also that in this context, the Wang's method performs better than the Denoeux's one. A new method generally satisfies better the second desirable property than the Denoeux's and Wang's methods. The only exception is the Example 1, where Wang's method provides somewhat more low value of σ than a new method. It is seen that the Wang's method better satisfies the second desirable property than the Denoeux's method. Summarizing, we can say that the proposed new method for the normalization of interval Dempster's rule of combination performs better than the known Denoeux's and Wang's methods. It is important that opposite to the Denoeux's and Wang's methods, there is no need for the solution of nonlinear programming task in the framework of a new method. This is its additional advantage.

4 Conclusion

A new method for normalization of interval Dempster's rule of combination within the framework of Dempster-Shafer theory of evidence based on the so-called "interval extended zero" method is developed. The two desirable intuitively obvious properties of normalization procedure are defined. The fists of them is based on the assumption that the sum of normalized interval weights should be an interval centered around 1 with a minimal width. The second desirable property is the remaining of ratios between the means of normalized intervals as close as possible to those of initial intervals before normalization. The advantages of a new method are illustrated with use of three numerical examples. It is shown that a new method performs better than the known Denoeux's and Wang's methods for combination of interval-valued belief structures as it provides the results with the properties which are close to the desirable ones. It is important that opposite to the Denoeux's and Wang's methods, there is no need for the solution of nonlinear programming task in the framework of a new method. This is its additional advantage.

References

1. Cleary, J.C.: Logical Arithmetic. Future Computing Systems 2, 125–149 (1987)
2. Denoeux, T.: Reasoning with imprecise belief structures. Internat. J. Approx. Reason. 20, 79–111 (1999)
3. Moore, R.E.: Interval analysis. Prentice-Hall, Englewood Cliffs (1966)
4. Sevastjanov, P., Dymova, L.: Fuzzy solution of interval linear equations. In: Procedings of 7th Int. Conf. Paralel Processing and Applied Mathematics, Gdansk, pp. 1392–1399 (2007)
5. Sevastjanov, P., Dymova, L.: A new method for solving interval and fuzzy equations: linear case. Information Sciences 17, 925–937 (2009)
6. Shafer, G.: A mathematical theory of evidence. Princeton University Press, Princeton (1976)
7. Wang, Y.M., Elhag, T.M.S.: On the normalization of interval and fuzzy weights. Fuzzy Sets and Systems 157 2456, 2456–2471 (2006)
8. Wang, Y.M., Yang, J.B., Xu, D.L., Kwai-Sang, C.: On the combination and normalization of interval-valued belief structures. Information Sciences 177, 1230–1247 (2007)

CI in General Game Playing - To Date Achievements and Perspectives

Karol Walędzik and Jacek Mańdziuk

Faculty of Mathematic and Information Science,
Warsaw University of Technology,
Pl. Politechniki 1, 00-661 Warsaw, Poland
{k.waledzik,j.mandziuk}@mini.pw.edu.pl

Abstract. Multigame playing agents are programs capable of autonomously learning to play new, previously unknown games. In this paper, we concentrate on the General Game Playing Competition which defines a universal game description language and acts as a framework for comparison of various approaches to the problem. Although so far the most successful GGP agents have relied on classic Artificial Intelligence approaches, we argue that it would be also worthwhile to direct more effort to construction of General Game Players based on Computational Intelligence methods. We point out the most promising, in our opinion, directions of research and propose minor changes to GGP in order to make it a common framework suited for testing various aspects of multigame playing.

1 Introduction

One of the most interesting areas of contemporary research on application of Artificial Intelligence (AI) and Computational Intelligence (CI) to mind games is the topic of multigame playing, i.e. development of agents able to effectively play any game within some general category being informed only about the rules of each of the games played. This poses a unique challenge to the AI community, as all the most successful game playing agents to date have been developed to achieve master level of play only in their specific games. Creating a system exhibiting high playing competency across a variety of previously unknown games would be a significant step in CI/AI research.

In the remainder of this paper we introduce the General Game Playing (GGP) framework and deal with the CI perspectives in GGP. We devote chapter 4 to analysis of possible machine learning approaches to GGP – identifying elements of existing programs (mainly AI-based) that can be incorporated into soft learning solutions and proposing a number of possible research directions. Finally, in chapter 5 we argue that GGP can easily become a universal multigame playing platform, useful in many research areas, even outside the context of the GGP tournament, as long as some necessary extensions are introduced into the standard.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 667–674, 2010.
© Springer-Verlag Berlin Heidelberg 2010

2 General Game Playing Competition

General Game Playing (GGP) [3] is one of several approaches to the multigame playing topic. It was proposed at Stanford University in 2005 in the form of General Game Playing Competition held annually at the National Conference for Artificial Intelligence [4]. General Game Players are agents able to interpret game rules described as a set of Game Description Language (GDL) [6] statements in order to devise a strategy allowing them to play those games effectively without human intervention.

The competition always includes a wide variety of games, both known previously and devised specifically for the tournament. Contestants should be prepared to deal with games of various complexity, varied branching factors and numbers of players, both cooperative and competitive.

2.1 Game Description Language

Game Description Language (GDL) [6] is used to describe the rules of the class of games playable within the GGP framework, i.e. finite, discrete, deterministic multi-player games of complete information. GDL describes games in a variant of Datalog. Game states are defined in terms of facts and algorithms for computing legal moves, subsequent game states, termination conditions and final scores for players are represented as logical rules.

3 GGP Competition Winners

In this section, selected most notable achievements in the field of GGP agents development are described. As it will become evident in the following sections, the winners of the first four editions of the contest relied on AI rather than CI methods. We believe, however, that elements of these successful AI solutions may be transferable to more CI-focused approaches, and these transferable aspects will be described in more detail in further chapters.

3.1 Cluneplayer

Cluneplayer [1], developed by James Clune, was the champion of the first and vice-champion of the second GGP tournament. It relies heavily on game domain specific observation that most mind games share a number of common concepts important in close-to-optimal play. Extracting definition of these crucial game features from game description should allow construction of a new simplified game in the form of a compound lottery based on the three core aspects of the original game: *expected payoff*, *control* (or mobility) measure and *game termination* probability (or game longevity). The expected outcome of thus created model approximates original game state evaluation.

3.2 Fluxplayer

Fluxplayer [9] was developed by S. Schiffel and M. Thielscher and proved superior to Cluneplayer in the second GGP championship. Similarly to Cluneplayer it depends on depth-first game tree search algorithm with widely known enhancements and automatically generated evaluation function based on fuzzy logic concepts. In each analyzed state approximate truth values of terminal and goal formulas are calculated and the program attempts to end the game whenever its goal is attained and avoid reaching terminal states when it would mean its loss.

3.3 CadiaPlayer

Cadiaplayer, developed by H. Finnsson and Y. Björnsson [2], is the first program that managed to win the GGP tournament two times in a row – in 2007 and 2008. Unlike earlier champions, it does not concentrate on advanced analysis of game description, relying, instead, on strong simulation-based game tree evaluation algorithm. Its operation is, in general case, based on Monte Carlo simulations enhanced by the UCT (Upper Confidence bounds applied to Trees) method. Cadiaplyer repeatedly plays partially random matches, gathering statistical data on each moves' relative strength. The move choice routine in each of the simulated matches is partially guided by the move quality data gathered so far.

4 Computational Intelligence Perspectives in GGP

So far all GGP champions relied on traditional AI approaches in the form of deterministic analysis of game description (in order to identify some well known patterns) and sophisticated game tree search algorithms. This is probably, to a large extent, caused by severely limited time allotted for learning before the actual match starts, while most CI learning processes require repetitive and time-consuming learning patterns presentation or environment sampling.

Even though current tournament rules make it difficult for an agent relying on Computational Intelligence methods to compete against deterministic symbolic approaches and/or brute-force game tree analysis algorithms in the competition itself, GGP-like environment with increased learning time limits can be used as a common test bed for various approaches to multigame playing agent implementation. In the following sections we intend to present our view on the promising research directions and concepts in this area.

Most perfect-information deterministic game playing agents make use of various minimax algorithms, and so far there is no evidence that this approach should be unsuitable for multigame playing. The most difficult (and interesting) part of minimax search-based agent development is construction of its game state evaluation function. There are many possible representations of this heuristic, but two of them have gained most popularity: linear combination of selected state features and artificial neural networks. In any case, the input game features should first be identified. Before presenting our solutions to those questions, we want, however, to take a peek at a CI-based GGP agent, which we think may provide inspiration for further research in the field of multigame playing with machine learning methods.

4.1 nnrg.hazel

nnrg.hazel [8] is one of the applications that suggest that CI-based approaches may yet prove successful in GGP-like problems. It was developed by J. Reisinger, E. Bahçeci, I. Karpov and R. Miikkulainen and came 6th (out of 12 contestants) in 2006 GGP Competition (5th in the preliminary rounds).

nnrg.hazel makes use of minimax game tree search method based on alpha-beta pruning algorithm with a depth limit of one ply. Game state evaluation function is represented as an artificial neural network (ANN). It is generated by a co-evolutionary method, evolving network topology and connection weights simultaneously. This is done using a method called NeuroEvolution of Augmenting Topologies (NEAT) [10]. It starts with the simplest possible fully-connected network with input and output layers only, which is then incrementally complexified to incorporate gradually more complex concepts without forgetting the knowledge already acquired.

The most obvious weakness of nnrg.hazel is its lack of any 'intelligent' ANN input data generation algorithm. Instead, the agent simply relies on random projection of game state features onto a 40-node ANN input layer. Improvements in this area, described more closely in the following section, seem to be the most obvious path of further development of NEAT-based General Game Players.

4.2 Game State Features Identification in Existing Solutions

While nnrg.hazel is able to achieve surprisingly promising results with only random projection of state description facts onto the evaluation function input vector, even its authors admit that it can be expected to fare much better with better input features generation routines. The input vector of a state evaluator should ideally include intelligently selected both static (e.g. pieces counts) and dynamic (e.g. mobility) features of the game state.

The first widely cited and influential paper on game features identification and generation was published by G. Kuhlmann, K. Dresner and P. Stone [5]. Their approach consists in syntactical analysis of game description (supported by simple simulations) in search for a set of general patterns, such as *successor relations* (inducing ordering), *counters* (incrementing in each time step according to successor relation), two-dimensional *boards*, *markers* (occupying cells on a board) and *pieces* (markers that can exist at at most one cell on a board at a time). Having identified these structures, it is possible to easily create a set of game features such as Manhattan distances between pieces or number of occurrences of markers.

Cluneplayer's game state features identification process is to some extent similar. A set of candidate features contains initially all expressions found in the game description. Additional features are then generated by automatic discovery of possible constants that can be substituted for all the variables in these expressions. Dedicated analysis algorithm can furthermore identify constant values that are, in some way, 'special' (i.e. differ significantly from the rest). Afterwards, Cluneplayer attempts to impose interpretations on the resulting features. The three available interpretations are *solution cardinality*, *symbol distance* and *partial solution*.

Fluxplayer takes the concept even further by replacing syntactic analysis of game description with exploration of semantical properties of rules definitions. That way higher level concepts can be detected more easily and with greater confidence. Fluxplayer evaluation function may make use of structures such as successor and order relations.

4.3 Constructing Game State Features in CI-Based Solutions

All the above-mentioned approaches have proved relatively successful as part of AI programs and we think that they may also be utilized in CI-based solutions for creating input vectors for evaluation functions in any form. What is worth noting, is that the patterns identified by the described applications generally fall into two categories:

1. universal logical predicates (e.g. successor and order relations) and expressions explicitly provided in game rules;
2. 'real-world' mind game specific predicates (e.g. boards, pieces).

We believe that truly universal and purely CI-based GGP agent should not include any of the latter, as there is no inherent reason stemming from the problem specification to assume that, e.g., the concept of board should be generally applicable.

In many cases the number of identified game features will be too big to directly include them all in the evaluation function (whatever its form). In such cases, some selection or dimensionality-reduction mechanism will be required. This task can be accomplished by deterministic methods; they may,

The simplest approach might be parallel construction of several state evaluators with varied input features sets and occasional comparison to decide which of them should be maintained in the pool of candidate solutions and which should be discarded. New candidate features sets can be generated randomly or via modification of existing sets depending on training results, e.g. utilizing sensitivity analysis.

4.4 Soft Learning in GGP Research Direction Proposals

Although the specifics of multigame playing application must differ significantly from the single-game programs, we believe that much of the knowledge gathered during development of single-game learning agents is transferable to the multigame case. One of the distinguishing aspects of GGP agents is that game tree traversal and identification of legal moves may prove much more time-consuming than in the case of single-game programs, since in GGP both these operations require costly theorem proving. Depending on the detailed setup of the learning approaches, this fact may make some methods less useful than others. Particularly, coevolutionary training schemes may turn out to be slower than expected.

We believe that one of the approaches that may be effective in terms of game tree traversal cost might be the layered learning scheme described, in the context of evolutionary algorithm, in [7]. The method requires dividing the game into several disjoint stages and gradually creates evaluation function applicable to all of them. First, a number of end-game positions are generated via random play. These positions are analyzed with minimax algorithm without evaluation function with the expectation that the search will reach terminal states in most of the cases; the rest will have neutral value assigned. This way a training set of game states with their evaluations is generated and any supervised learning method can be used to construct the heuristic evaluation function. A number of new positions from an earlier game stage is then generated and evaluated using minimax search with depth limit of at least game stage length and the heuristic evaluation function from previous step. The new training set is used for further improvement of evaluation function. This process is repeated until the game tree root is reached.

GGP covers a very wide spectrum of possible games. Many machine learning algorithms can prove very sensitive to the choice of their parameters and game-specific features. Ideal General Game Player should, therefore, be able to tune its learning patterns to the problem at hand. We propose two basic approaches allowing to achieve this goal.

Firstly, the agent's training scheme could involve tuning the learning algorithm's steering parameters. This task could be performed by employing some kind of an evolutionary approach analogical to the one proposed for input features selection. A population of candidate solutions would be trained with separate sets of steering parameters. Their training results would occasionally be compared in order to identify the most promising individuals. The weakest would then be disposed of and the strongest reproduced into the next population. At some point the parameters would usually have to be frozen, and training of single candidate would continue to further optimize the evaluation function until some stop condition (e.g. running out of time) was met.

More sophisticated applications might take the same idea further and try to not only choose the best steering parameters but the training algorithm itself, comparing, for instance, various representations of evaluation function and/or learning schemes. The principles of the selection process would remain similar to those described in the previous paragraph, with parallel application of all the schemes and occasional comparison of their quality to abandon the weakest solutions. Alternatively, a whole ensemble of evaluation functions could be constructed along with rules describing how their results should be combined depending, for instance, on game phase.

5 GGP Framework Extension Proposals

5.1 Opponent Modeling

GGP, in its current form, does not offer enough information to seriously attempt long-term modeling of individual opponents. Contestants typically resort to one of popular simplifications: assuming that other players will implement decision process analogical to the playing agent's one, treating other players as opponents attempting to minimize the agent's score or simply considering random reactions of other players.

GGP framework can, however, easily be extended to include unique identifiers of players, thus forming a useful opponent modeling test bed. Game playing agents would then be able to gather knowledge about individual players' behavior and transfer it from match to match. Full opponent modeling scheme should also attempt to identify cross-game elements of player style, such as its aggressiveness (tendency to play risky moves, that may yield both high rewards and losses, depending on other players' responses), ability of long-term, strategic planning, probability of employing mobility strategies and so on. Identified traits could then be incorporated into game simulation and minimax tree analysis algorithms to better predict opponents' behavior.

5.2 Knowledge Transfer

Analogically to the case of opponent modeling, cross-game knowledge transfer is a goal very difficult to pursue in the current form of GGP framework. Game playing agents are not provided with game names or categorization and, during the tournament, even the GDL constants and predicate identifiers are garbled to devoid the programs of any lexical clues to their meaning. In this context, knowledge transfer would require very sophisticated game description analysis able to identify and extract patterns common to the games.

GGP framework communication protocols could, however, yet again be slightly modified in order to create a cross-game knowledge transfer testing environment, in which agents are provided with game names and categorization and GDL atoms with same names represent analogical data in varied games. In the simplest approach, game agent could attempt to store metaparameters that proved most successful in evaluation function learning for each game and attempt to select one of those sets whenever a new game is encountered (relying on game similarities). In case of games classified into the same category and sharing a significant number of concepts, the learning phase could gain a head-start by starting from solutions based on evaluators successful in similar games, instead of learning from scratch.

6 Summary

Although not new, the concept of multigame playing remains a very interesting and still little explored research area. We believe that it is well suited for application and comparison of various Computational Intelligence methods. One of the important factors that could help this research area flourish, would be a common framework defining class of games along with their description language that should be understood by the game playing agents and allowing direct comparison of various approaches. General Game Playing competition offers exactly that.

Although so far the tournaments seem better suited for symbolic approaches, we argue that it is possible to create a reasonable CI-based GGP player. In this paper, we have discussed the key aspects of developing CI-based General Game Players, identified the elements of the existing applications that can be transferred to machine learning approaches and proposed several promising research directions for the CI community. Implementation and validation of some of these proposals is, at the moment, a work in progress.

At the same time, we have pointed out that some aspects of multigame playing, such as opponent modeling and cross-game knowledge transfer, are prohibitively hard to tackle in the current form of the GGP framework. We therefore propose slight modifications to its definition so that it can become a standard multigame playing platform for testing and comparison of various approaches, even outside the scope of the annual championship.

References

1. Clune, J.: Heuristic evaluation functions for General Game Playing. In: Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (AAAI 2007), Vancouver, BC, Canada, pp. 1134–1139. AAAI Press, Menlo Park (2007)
2. Finnsson, H., Björnsson, Y.: Simulation-based approach to General Game Playing. In: Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI 2008), Chicago, IL, pp. 259–264. AAAI Press, Menlo Park (2008)
3. General Game Playing website, http://games.stanford.edu/
4. Genesereth, M., Love, N.: General Game Playing: Overview of the AAAI Competition (2005), http://games.stanford.edu/competition/misc/aaai.pdf
5. Kuhlmann, G., Dresner, K., Stone, P.: Automatic heuristic construction in a complete General Game Player. In: Proceedings of the Twenty-First AAAI Conference on Artificial Intelligence (AAAI 2006), Boston, MA, pp. 1457–1462. AAAI Press, Menlo Park (2006)
6. Love, N., Hinrichs, T., Haley, D., Schkufza, E., Genesereth, M.: General Game Playing: Game Description Language Specification (2008), http://games.stanford.edu/language/spec/gdl_spec_2008_03.pdf
7. Mańdziuk, J., Kusiak, M., Walędzik, K.: Evolutionary-based heuristic generators for checkers and give-away checkers. Expert Systems 24(4), 189–211 (2007)
8. Reisinger, J., Bahçeci, E., Karpov, I., Miikkulainen, R.: Coevolving strategies for general game playing. In: Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG 2007), Honolulu, Hawaii, pp. 320–327. IEEE Press, Los Alamitos (2007)
9. Schiffel, S., Thielscher, M.: Automatic Construction of a Heuristic Search Function for General Game Playing. In: Seventh IJCAI International Workshop on Nonmonotonic Reasoning, Action and Change (NRAC 2007) (2007)
10. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation 10(2), 99–127 (2002)

Soft Computing Approach to Discrete Transport System Management

Tomasz Walkowiak and Jacek Mazurkiewicz

Institute of Computer Engineering, Control and Robotics, Wroclaw University of Technology, ul. Janiszewskiego 11/17, 50-372 Wroclaw, Poland
Tomasz.Walkowiak@pwr.wroc.pl, Jacek.Mazurkiewicz@pwr.wroc.pl

Abstract. A serious problem in the transport system analysis is to find a suitable methodology for modelling the management presented in real systems. It is hard to create an "intelligent" algorithm of dispatcher - an algorithm giving significantly better results from pure random algorithms. In the paper we propose a multilayer perceptron approach to solve this problem. The neural network is learned using a genetic algorithm. A fitness function is defined by business service requirements of discrete transport system (DTS). The proposed approach is based on modelling and simulating of the system behaviour. Monte Carlo simulation is a tool for DTS performance metric calculation. No restriction on the system structure and on a kind of distribution is the main advantage of the method. The system is described by the formal model, which includes reliability and functional parameters of DTS. The proposed, novelty approach can serve for practical solving of essential management problems related to an organization of transport systems.

1 Introduction

Management of a large transport system is not a trivial task. The transport systems are characterized by a very complex structure. The performance of the system can be impaired by various types of faults related to the transport vehicles, communication infrastructure or even by traffic congestion. The systems are often driven by a dispatcher - a person who allocates vehicles to the tasks introduced into system. The dispatcher ought to take into account different features which describe the actual situation of all elements of transport systems. The modelling of transport systems with dispatcher is not a trivial challenge. The most effective method is to use a time event simulation with Monte Carlo analysis [2]. It allows to calculate different system measures which could be a base for decisions related to administration of the transport systems. No restriction on the system structure and on a kind of distribution is the main advantage of the method. The paper presents an analysis of transport system working for the Polish Post regional centre of mail distribution (described in section 2). Base on which we have developed the transport system model presented in sections 3.1-3.4. The real transport system of Polish Post is managed by the set of timetables (section 3.5), which are composed and changed according to season of

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 675–682, 2010.
© Springer-Verlag Berlin Heidelberg 2010

the year, holidays, etc. We propose to substitute the legacy management system (section 3.5) by the soft computing management system (section 3.6). Our previous works [10], [9] showed that it is very hard to find an ”intelligent” algorithm of dispatcher - an algorithm which is giving results which significantly differ from the pure random algorithm. In many papers, i.e. [1] [5], it was shown that the evolutionary approach [3] to challenging science or engineering problem gives very promising results. In a paper [6] we have shown a usage of genetic algorithms to selection of a best set of rules for management of transport system (different from the system presented here). For the transport system presented here we propose a usage of neural networks learned by genetic algorithm methods [5] (section 3.7). The approach is verified (section 4) using a simulator developed by authors [7],[8].

2 Discrete Transport System in Polish Post

The analysed transport system is a simplified case of the Polish Post. The business service provided by the Polish Post is the delivery of mails. The system consists of a set of nodes placed in different geographical locations. Two kinds of nodes could be distinguished: central nodes *(CN)* and ordinary nodes *(ON)*. There are bidirectional routes between them. Mails are distributed among ordinary nodes by trucks, whereas between central nodes by trucks, by trains or by planes. The mail distribution could be understood by tracing the delivery of some mail from point A to point B. At first the mail is transported to the nearest ordinary node. Different mails are collected in ordinary nodes, packed in larger units called containers and then transported by vehicles scheduled according to a time-table to the nearest central node. In central node containers are repacked and delivered to appropriate (according to delivery address of each mail) central node. In the Polish Post there are 14 central nodes and more than 300 ordinary nodes. There are more than one million mails going through one central node within 24 hours. It gives a very large system to be modelled and simulated. Therefore, we have decided to model only a part of the Polish Post discrete transport system - one central node with a set of ordinary nodes. The income of mail containers to the system is modelled by a stochastic process. Each container has a source and destination address. The central node is the destination address for all containers generated in the ordinary nodes. In case of central node, there are separate random processes for each ordinary node. Each vehicle has a given capacity - maximum number of containers it can haul. Central node is a base place for all vehicles. The vehicle time-table consists of a set of routes (sequence of nodes starting and ending in the central node, time-moments of leaving each node in the route and the recommended size of a vehicle). The number of used vehicles and the capacity of them does not depend on temporary situation described by number of transportation tasks or by the task amount for example. The process of vehicle operation could be stopped at any moment due to a failure (random process). After the failure, the vehicle waits for a maintenance crew (if there are no available), is being repaired (random time) and after it continues its journey. The vehicle hauling a commodity is always fully loaded or taking the last part of the commodity if it is less than its capacity.

3 Formal Model of Discrete Transport System

3.1 Model Overview

The described in the previous section regional part of the Polish Post transport system with one central node and several ordinary nodes was a base for a formal model definition of the discrete transport system with central node (DTS). Users generate tasks which are being realised by the system. The task to be realised requires some services available in the system. A realisation of the service needs a defined set of technical resources. Moreover, the vehicles transporting mails between system nodes are steering by the management system. Therefore, we can model discrete transport system as a quadruple [8]:

$$DTS = \langle Client, BS, TI, MS \rangle \,, \tag{1}$$

where: *Client* - client model, *BS* - business service, a finite set of service components, *TI* - technical infrastructure, *MS* - management system.

3.2 Technical Infrastructure

During modelling of technical infrastructure we have to take into consideration functional and reliability aspects of the post transport system. Therefore, the technical infrastructure of DTS could be described by three elements:

$$TI = \langle No, V, MM \rangle \,, \tag{2}$$

where: *No* - set of nodes, *V* - set of vehicles, *MM* - maintenance model.

Set of nodes *(No)* consists of single central node *(CN)* and a given number of ordinary nodes *(ON_i)*. The distance between each two nodes is defined by the function:

$$distance : No \times No \to R_+ . \tag{3}$$

Each node has one functional parameter the mean (modelled by normal distribution) time of loading a vehicle:

$$loading : No \to R_+ . \tag{4}$$

Moreover, the central node *(CN)* has additional functional parameter: number of service points (in each ordinary node there is only one service point):

$$servicepoints : CN \to N_+ . \tag{5}$$

Each vehicle is described by following functional and reliability parameters:

- mean speed of a journey - $meanspeed : V \rightarrow R_{+}$,
- capacity - $capacity : V \rightarrow N_{+}$,
- mean time to failure - $MTTF : V \rightarrow R_{+}$,
- mean repair time - $MRT : V \rightarrow R_{+}$.

The traffic is modelled by a random value of vehicle speed and therefore the time of vehicle *(v)* going from one node *(n_1)* to the other *(n_2)* is given by a formula:

$$time(v, n_1, n_2) = \frac{distance(n_1, n_2)}{Normal(meanspeed(v), 0.1 \cdot meanspeed(v))} \tag{6}$$

where: *Normal* denotes a random value with the Gaussian distribution.

Maintains model *(MM)* consists of a set of maintenance crews which are identical and unrecognized. The crews are not combined to any node, are not combined to any route, they operate in the whole system and are described only by the number of them. The time when a vehicle is repaired is equal to the time of waiting for a free maintains crew (if all crews involved into maintenance procedures) and the time of a vehicle repair which is a random value with the Gaussian distribution: $Normal(MRT(v), 0.1 \cdot MRT(v))$.

3.3 Business Service

Business service *(BS)* is a set of services based on business logic, that can be loaded and repeatedly used for concrete business handling process. Business service can be seen as a set of service components and tasks, that are used to provide service in accordance with business logic for this process. Therefore, *(BS)* is modelled a set of business service components *(sc)*:

$$BS = \{sc_1, ..., sc_n\}\,,\ n = length(BS) > 0, \tag{7}$$

the function *length(X)* denotes the size of any set or any sequence X.

Each service component in DTS consists of a task of delivering a container from a source node to the destination one.

3.4 Client Model

The service realised by the clients of the transport system are sending mails from some source node to some destination one. Client model consists of a set of clients *(C)*. Each client is allocated in some node of the transport system:

$$allocation : C \rightarrow No. \tag{8}$$

A client allocated in an ordinary node generates containers (since, we have decided to monitor containers not separate mails during simulation) according to the Poisson process with destination address set to ordinary nodes. In the central node, there is a set of clients, one for each ordinary node. Each client generates containers by a separate Poisson process and is described by intensity of container generation:

$$intensity : C \rightarrow R_{+}. \tag{9}$$

The central node is the destination address for all containers generated in ordinary nodes.

3.5 Legacy Management System

The management system *(MS)* of the DTS controls the operation of the vehicle. In a case of Polish Post DTS consists of a sequence of routes:

$$MS = \langle r_1, r_2, ..., r_n \rangle \, . \tag{10}$$

Each route is a sequence of nodes starting and ending in the central node, times of leaving each node in the route *(t_i)* and the recommended size of a vehicle (size):

$$r = \langle CN, t_0, n_1, t_1, ..., n_m, t_m, CN, size \rangle \, , \tag{11}$$

$$v_i \in No - \{CN\} \, , \; 0 \leq t_0 < t_1 < ... < t_m < 24h \tag{12}$$

The routes are defined for one day and are repeated each day. The management system selects vehicles to realise each route in random way, first of all vehicles (among vehicles available in central node) with capacity equal to recommended size are taken into consideration. If there is no such vehicle, vehicles with larger capacity are taken into consideration. If still there is no vehicle fulfilling requirements vehicle of smaller size is randomly selected. If there is no available vehicle a given route is not realised.

3.6 Soft Computing Management System

As it was mentioned in the introduction we propose a neural network based replacement of the legacy management system. The system consists of a multilayer perceptron to decide if and where to send trucks. The input to the neural network consist of:

$$in = \langle onc_1, onc_2, ..., onc_{non}, cnc_1, cnc_2, ..., cnc_{non}, nfv \rangle \, , \tag{13}$$

where: *non* - number of ordinary nodes, *onc_i* - number of containers waiting for delivery in *i*-th ordinary node, *cnc_i* - number of containers waiting for delivery in the central node with destination address set to i-th ordinary node, *nfv* - number of free vehicles in the central node.

Each output of the network corresponds to each ordinary node:

$$nnout = \langle out_1, out_2, ..., out_{non} \rangle \, , \tag{14}$$

The output of the network is interpreted as follows (for sigmoid function used in output layer):

$$j = argmax_{i=1...non} \{out_i\} \tag{15}$$

If out_j is greater than 0.5 send a vehicle to node j else do nothing. If there are more vehicles available in the central node, the largest vehicle that could be fully loaded is selected. If there are available several trucks with the same capacity selection is done randomly. The neural network decision (send a truck or not and where the truck should be sent) are taken in given moments in time. These moments are defined by following states of the system:

- the vehicle comes back to the central node and is ready for the next trip;
- if in central node there is at least one available vehicle and the number of containers of the same destination address is larger than the size of the smallest available vehicle.

3.7 Neural Network Learning

The neural network used in the management system requires a learning process that will set up the values of its weights. The most typical learning in the case of multilayer perceptron is the back propagation algorithm. However, it cannot be used here since it is impossible to state what should be the proper output values of the neural network. Since it is hard to reconcile what are the results of a single decision made by the management system. Important are results of the set of decisions. Since the business service realised by transport system is to move commodities without delays, the neural network should take such decisions that allows to reduce delays as much as possible. To train neural network to perform such task we propose to use genetic algorithm [3]. Similar approach to training neural network is applied in case of computer games [5]. The most important in case of genetic algorithm is a definition of the fitness function. To follow business service requirements of transport system we propose following definition of the fitness function calculated for a given neural network after some time *(T)* (therefore after a set of decisions taken by neural network):

$$fitness(T) = \frac{N_{ontime}(0,T) + N_{ontimeinsystem}(T)}{N_{delivered}(0,T) + N_{insystem}(T)}. \tag{16}$$

It is a ratio of on-time containers (delivered with 24h and being in the system but not longer then 24h) to all containers (that already delivered $N_{delivered}(0,T)$ and still being presented in the system $N_{insystem}(T)$).

4 Experiments and Results

We propose for the case study analysis a simple DTS. The system is composed of one central node (marked as *CN*) and three ordinary nodes (marked as *ON1*, *ON2* and *ON3*). The length of roads was set to 80 km for roads between central node and each of ordinary nodes. The intensity of generation of containers for all destinations was set to 4 per hour. The vehicles speed was modelled by Gaussian distribution with 45km/h of mean value and 4.5km/h of standard deviation. The average loading time was equal to 5 minutes. A capacity of each of vehicles was set to 10 containers. The MTTF of each vehicle was set to 2000. The average repair time was set to 5h (Gaussian distribution). For the purpose of simulating DTS we have developed a simulator [7],[8] using Parallel Real-time Immersive Modelling Environment (PRIME)[4] simulation core.

To transport all generated in the system containers the time-table has to be setup. For the described example it was designed manually in a manner to allow to transport all containers (with 10% overhead) and 15 minutes of break between each drive. It resulted in a usage of 10 vehicles. The soft computing management system trained by genetic algorithm was able to achieve the (16) metric values equal almost to 1 similarly to the legacy management system (with a time-table). To test the soft computing management system more deeply we analysed the transport system in a case of some system resource degradation resulting in the transport system performance degradation [7]. During a training of neural networks in some of generations (with a probability of 0.1) a critical situation occurs (for all population members). The system is analysed for a 10 days of normal performance and then for a given number of days (1-10, selected by a random generator) 50% of vehicles are not available. After these days the number of vehicles backs to normal vales. To illustrate the achieved results we are using a metric similar to (16). The acceptance ratio [7] is a ratio of on-time containers to all containers within a 24h time period. Therefore a sequence of time moments (when the metric is calculated has to be set - a midnight of each day was used). The results for the legacy and soft computing algorithm for some exemplar critical situation is presented in Fig. 1. As it could be expected the acceptance ratio in day 10 is starting to drop down to 0.63 and when all trucks are available again (on the day 13th) is slowly enlarging. Important is the fact that a transport system with soft computing management is able to back to a normal performance faster than that with a time-table.

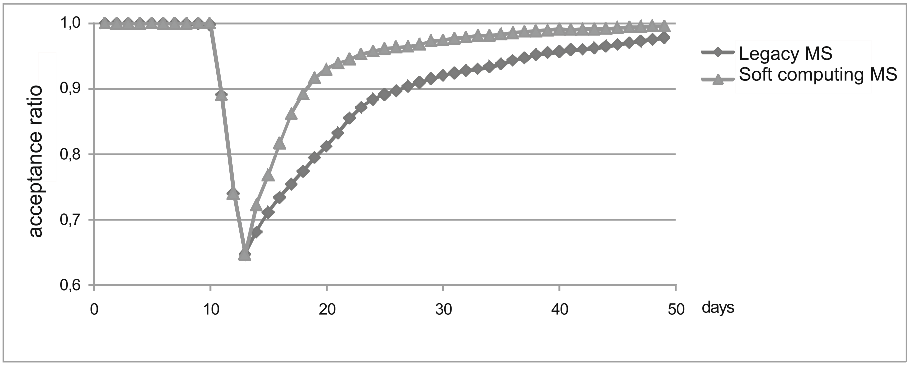


Fig. 1. Results - legacy and soft computing management systems comparison

5 Conclusion

Summarising, we have proposed the soft computing approach to management of discrete transport systems. The multilayer perceptron learned using genetic algorithm successfully substitutes the classic set of time-tables. Our solution seems to be very useful in case of critical situations, which could not be included in time-tables. Also Monte-Carlo approach as a device for event simulation can be evaluate very promising to relax the strict Markov assumptions. The presented approach could be used as a foundation for a new methodology of the functionality and economic analysis of transport systems with dispatcher, which is much closer to the practice experience. We plan to test the approach on the larger transport system which models local post distribution in Dolny Slask [7].

References

1. Corne, D.W., Fogel, G. (eds.): Evolutionary Computation in Bioinformatics. Morgan Kaufman Publishers, San Francisco (2003)
2. Fishman, G.: Monte Carlo: Concepts, Algorithms and Applications. Springer, Heidelberg (1996)
3. Koza, J.: Genetic Programming: On The Programming of Computers by Means of Natural Selection. MIT Press, Cambridge (1992)
4. Liu, J.: Parallel Real-time Immersive Modelling Environment (PRIME), Scalable Simulation Framework (SSF), User's manual. Colorado School of Mines Department of Mathematical and Computer Sciences (2006), http://prime.mines.edu/
5. Olofsson, J.F., Andersson, W.: Human-like Behaviour in Real Time Strategy Games - An Experiment with Genetic Algorithms, Blekinge Institute of Technology (2003)
6. Walkowiak, T., Mazurkiewicz, J.: Genetic Approach to Modeling of a Dispatcher in Discrete Transport Systems. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 479–488. Springer, Heidelberg (2006)
7. Walkowiak, T., Mazurkiewicz, J.: Analysis of critical situations in discrete transport systems. In: Proceedings of International Conference on Dependability of Computer Systems, Brunow, Poland, June 30-July 2, pp. 364–371. IEEE Computer Society Press, Los Alamitos (2009)
8. Walkowiak, T., Mazurkiewicz, J.: Event simulation for reliability and functional analysis discrete transport systems. Reliability and statistics in transportation and communication. In: Kabashkin, I.V., Yatskin, I.V. (eds.) RelStat: Proceedings of the 9th International Conference, October 21-24, pp. 63–71. Riga: Transport and Telecommunication Institute, Riga (2009)
9. Walkowiak, T., Mazurkiewicz, J.: Reliability and Functional Analysis of Discrete Transport System with Dispatcher. In: Advances in Safety and Reliability - ESREL 2005, pp. 2017–2023. Taylor & Francis Group, Abington (2005)
10. Walkowiak, T., Mazurkiewicz, J.: Simulation Based Management and Risk Analysis of Discrete Transport Systems. In: IEEE TEHOSS 2005 Conference, Poland, pp. 431–436 (2005)

Crowd Dynamics Modeling in the Light of Proxemic Theories

Jarosław Wąs

Institute of Automatics
AGH University of Science and Technology
al. Mickiewicza 30, 30-059 Kraków, Poland
jarek@agh.edu.pl

Abstract. The application of the theory of proxemics brings a promising perspective to microscopic motion modeling in pedestrian dynamics. Combining an agent-based approach and spatial context make it possible to simulate crowd in different classes of situations. The article discusses certain aspects of proxemics theory and the possibility of using the Social Distances model for different classes of situations. Also, an idea of using specialized borderline cells is introduced, which enables more precise space representation.

1 Introduction

Over the last years, we could observe growing interest in the microscopic approach to modeling of pedestrian dynamics. In order to understand pedestrian movement from a bird's eye view, we have to take into account microscopic interactions between pedestrians; their aims and the influence of familiarities have to be taken into consideration. As a result, the actual methodology of pedestrian dynamics simulation moves away from cellular automata or social forces (molecular dynamics) towards more sophisticated, agent-based approaches [2], [4], [1].

2 Theory of Proxemics

Studies of how people use physical space in interpersonal interaction are based on E.T. Hall's proxemics theory [5], [6]. Three components of his theory seem to be the most important [7] [13]

- proxemics is connected with interpersonal interactions,
- interactions are defined in specific spatial context, defined by four informal areas surrounding each person: *intimate*, *personal*, *social* and *public*.
- behaviors are largely learned or culturally determined rather then entirely dictated by innate biological or psychological processes.

The sum of the behaviors involved in interpersonal interaction may be bewilderingly complex, but a relatively small subset of them has been shown to be exceptionally significant.

Proxemics defines different types of space [9],[10]:

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 683–688, 2010.
© Springer-Verlag Berlin Heidelberg 2010

Fixed-feature space. This comprises things that are immobile, such as walls and territorial boundaries.

Semifixed-feature space. This comprises movable objects, such as pieces of furniture.

Movement space. This comprises space available for pedestrian traffic.

Informal space. This comprises the personal space around the body, that travels around with a person as he/she moves, and that determines the personal distance among people.

Proxemics also classifies spaces as either sociofugal or sociopetal. The terms are analogous to the words "centrifugal" and "centripetal" [8].

3 Applying Theory of Proxemics in Crowds Modeling

Theory of proxemics in crowd dynamics was introduced in [11] as the *Social Distances* model with extensions in [12]. The model is still being developed. It is based on an elliptic representation of pedestrians placed on a square lattice. One of the main assumptions in the model is that social areas are elliptical and asymmetrical (Fig 1). Configuration in the front of a pedestrian, in majority of situations, is more important for this pedestrian than configuration behind them. Thus, the model has to distinguish the front and the back of a person.

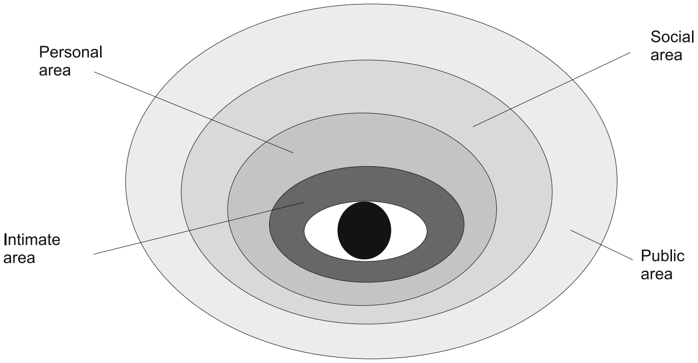


Fig. 1. The author's interpretation of proxemics theory - asymmetrical and elliptical social distances. The upper part marks the front of a person.

The rules of proxemics are slightly different for various classes of pedestrian situations, such as: freeway traffic, controlled evacuation, competitive evacuation or panic. In static, *passive* situations (e.g. waiting) the pedestrians are likely to act according to social distances principles. In dynamic, *active* situations (e.g. looking for something or evacuating) pedestrians are oriented on defined aims and they sometimes violate social distances of others. Thus, pedestrians-agents in the model have these two available states: *active* and *passive*.

3.1 Spatial Aspects of the Model

The model is based on a 2-dimensional non-homogeneous cellular automaton (can also be interpreted as MAS). In the model, space is represented as a lattice with square cells. The size of each cell equals 0.25 m, while the size of each ellipsis equals: semimajor axis equals a = 0.225 m and semiminor axis b = 0.135 m connected the average size of a person according to WHO data.

Parameter ϵ in the presented model describes pedestrian allocation in space, using allowed and forbidden configurations. Thanks to this parameter crowd compressibility can be taken into consideration.

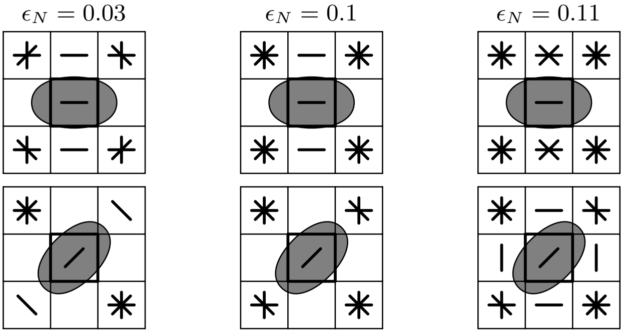


Fig. 2. Allowed neighborhood configurations for different tolerance parameters

3.2 Specialized Types of Cells in the Model Lattice

Previous versions of the model have not solved the question how to represent boundaries of movement space. Thus, a special class of border cells is proposed now to resolve this issue. These cells are *borderlines* between walls and movement space. Their introduction makes it possible to eliminate some spatial problems with model calibration.

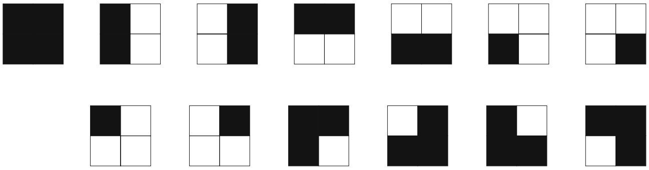


Fig. 3. Each borderline cell is divided into four sections. Sections marked black are interpreted as walls or obstacles. Sections marked white represent movement space.

3.3 Movement Rules

Each agent is represented as an object on a square lattice [3]. An agent posses a set of attributes as: a list of aims, desired velocity etc. A current aim –attractor is defined on a base of a cost function. An agent heads towards the aim – attractor (for instance exit, cash desk etc) according to a gradient of potential field 4. The cost function is defined below:

$$Fcost_{ij} = w_d d_i(a_j)) + w_\rho \rho(a_j) \tag{1}$$

$$\rho(a_j) = \frac{1}{|N(a_j,\lambda)|} \sum_{k=1}^{|N(a_j,\lambda)|} s(c_k) \tag{2}$$

where:

$Fcost_{ij}$ - a cost of decision to use an attractor a_j by i-th pedestrian

$i = 1, 2, ..., |L|$ - index of pedestrians

$j = 1, 2, ..., |A|$ - index of attractors

$d_i(a_j)$ - normalized distance of i-th pedestrian to j-th attractor, $d_i(a_j) \in [0; 1]$

$\rho(a_j)$ - crowd density around j-th attractor defined by (2), $\rho(a_j) \in [0; 1]$

w_d, w_ρ - criterions weights, $w \in [0; 10]$

$N(a_j, \lambda)$ - Moore neighborhood of a radius λ

$s(c_k)$ - state of k-th cell (where 0 means vacant cell and 1 means occupied cell)

3.4 Implementation

The model is implemented in C++ using standard MFC library.

Figure 4 illustrates a situation of large room evacuation situation. Pedestrians are represented by ellipses and are all marked gray. The movement being realized is heading towards the exit (also marked gray). This means that all pedestrians are in an *active* state. It has been stressed that only classes of cells belongs completely to the movement space (marked white) have a given value of potential field. Borderline cells only play complementary function and they do not have any potential values. Pedestrians move towards a decreasing gradient of the potential field and they also take into account a set of allowed/forbidden configurations in their movement rules.

Figures 5 illustrates a situation of passenger flow in a municipal communication vehicle. A group of pedestrians marked gray is heading towards the seats (also gray), while other pedestrians marked black recede, because their social

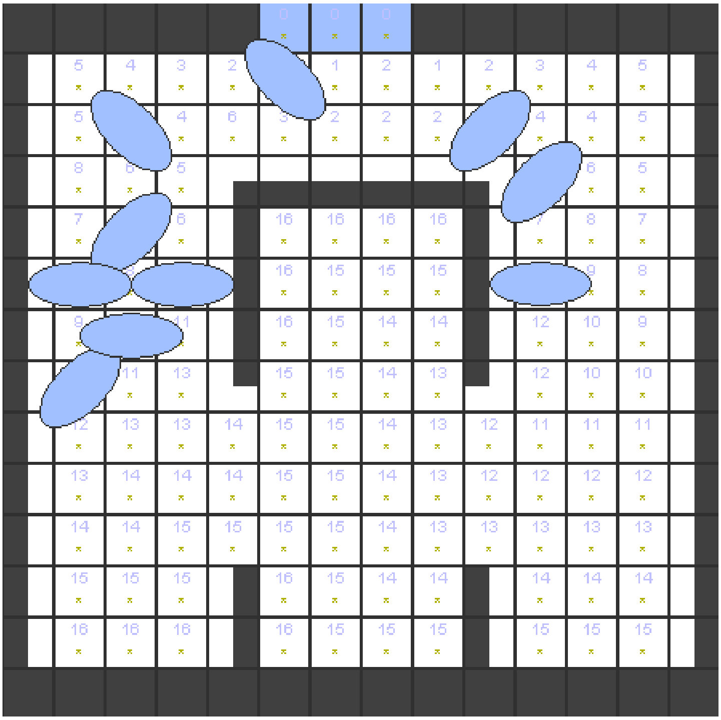


Fig. 4. Large room evacuation - different borderline cells are used for detailed topology representation

distance is violated. Gray agents are in an *active* state, while black ones are in a *static* state.[1]

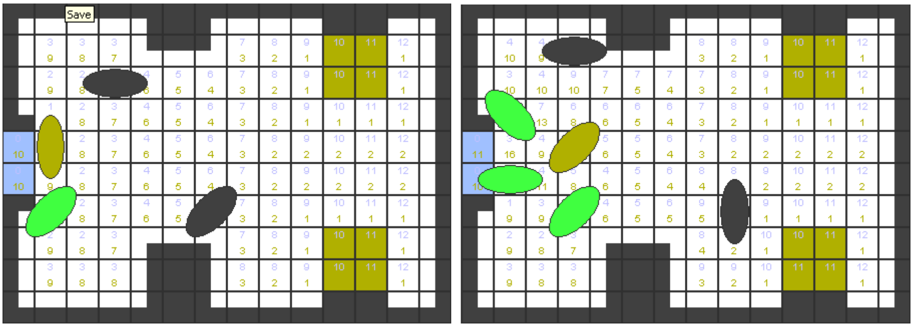


Fig. 5. Two phases of social distances operating

[1] Light green agents represent another *active* agents.

4 Concluding Remarks

The model of Social Distances is developed and consequently prepared for real-world simulations. The necessity of its calibration and validation requires transferring a generic idea into concrete solutions. As it was presented in the previous section, the model can be used not only for free pedestrian traffic, but also for evacuation situations. During evacuation situations simulated population is only (in general) in an *active* state, while during freeway traffic population consists of both: *active* and *passive* agents.

The article discusses some spatial issues of Social Distances model. The introduction of specialized types of borderline cells is proposed, which make it possible to represent space in a more accurate way, with the required details.

Acknowledgment

This research is financed by the Polish Ministry of Science and Higher Education, Project no: N N516 228735.

References

1. Bandini, S., Manzoni, S., Vizzari, G.: Situated Cellular Agents a Model to Simulate Crowding Dynamics. Special Issues on Cellular Automata, 669–676 (2004)
2. Dijkstra, J., Jessurun, A.J., Timmermans, H.: A Multi-Agent Cellular Automata System for Visualising Simulated Pedestrian Activity. In: Proceedings of ACRI, pp. 29–36 (2000)
3. Dudek–Dyduch, E., Wąs, J.: Knowledge Representation of Pedestrian Dynamics in Crowd: Formalism of Cellular Automata. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 1101–1110. Springer, Heidelberg (2006)
4. Gloor, C., Stucki, P., Nagel, K.: Hybrid Techniques for Pedestrian Simulations. In: Sloot, P.M.A., Chopard, B., Hoekstra, A.G. (eds.) ACRI 2004. LNCS, vol. 3305, pp. 581–590. Springer, Heidelberg (2004)
5. Hall, E.T.: The Silent Language. Garden City, New York (1959)
6. Hall, E.T.: The Hidden Dimension. Garden City, New York (1966)
7. Ickinger, W.J.: Behavioral game methodology for the study of proxemic behavior. Yale University, New Haven (1982)
8. Lawson, B.: Sociofugal and sociopetal space. In: The Language of Space, pp. 140–144. Architectural Press (2001)
9. Littlejohn, S., Foss, K.: Theories of Human Communication. In: Thomson Wadsworth Communication, pp. 107–108 (2005)
10. Low, S.M., Lawrence-Zúniga, D.: The Anthropology of Space and Place: Locating Culture. Blackwell Publishing, Malden (2003)
11. Wąs, J., Gudowski, B., Matuszyk, P.J.: Social Distances Model of Pedestrian Dynamics. In: El Yacoubi, S., Chopard, B., Bandini, S. (eds.) ACRI 2006. LNCS, vol. 4173, pp. 492–501. Springer, Heidelberg (2006)
12. Wąs, J.: Multi-agent Frame of Social Distances Model. In: Umeo, H., Morishita, S., Nishinari, K., Komatsuzaki, T., Bandini, S. (eds.) ACRI 2008. LNCS, vol. 5191, pp. 567–570. Springer, Heidelberg (2008)
13. Proxemics, `http://en.wikipedia.org/wiki/Proxemics`

The Use of Psycholinguistics Rules in Case of Creating an Intelligent Chatterbot

Sławomir Wiak[1] and Przemysław Kosiorowski[2]

[1] Institute of Mechatronics and Information Systems,
Technical University of Lodz, Poland
swiak@p.lodz.pl
[2] Institute of Mechatronics and Information Systems,
Technical University of Lodz, Poland
pkosiorowski@o2.pl

Abstract. This paper presents the use of psycholinguistics [1] rules in the case of creating an intelligent Chatterbot. Synonyms [2], hyponyms [2] and hypernyms [2] will be defined and implemented as the rules while the database of Chatterbot being created. The algorithm thanks to which it is possible to get better accuracy in generated and searched phrases by Chatterbot is presented in the paper. Moreover, authors have also made comparative analysis of existing platforms and the Chatterbot programming languages with the proposed system.

Keywords: chatbot, chatbot programming, artificial intelligence, data base, intelligent systems.

1 Introduction

Joseph Weinzenbaun [8] from Massachussetts Institute of Technology has created first code dedicated to Internet chat (chatterbot) called Eliza in 1966. This chatterbot had gained popularity a few years later when the Internet became more common. Since that time a few interesting applications have been created, i.e. Snikers, Dianthus, Denise, Fido-Interactive, etc.

The general target of chatterbots applications is to replace the humans by dedicated computer applications, while communication skills and technologies being developed. The further development of chatterbots is based on equipping these programs in new mechanisms and configurations, leading to dedicated chatterbots to each topic or discipline. The largest corporations such as Volkswagen, IKEA, ErgoHestia have began to use this type of chatterbots as information bots. We could even stress, that such chaterbots are commonly used in customers service systems, and call centers instead workers.

The current progress in chatterbot technologies enables to create intelligent bots with almost no difference between human communication, and computer communication as well. Generally, it is very hard target to reach such a level of communication. The main barrier to overcame lays in the lack of semantic meaning and understanding of the phrases. Moreover, searching the database to find corresponding either word or phrase is time consuming procedure. This inaccuracy could be decreased by implementing better algorithms and better suitable linguistic rules.

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 689–697, 2010.
© Springer-Verlag Berlin Heidelberg 2010

Psycholinguistic rules used in creating of intelligent Chatterbot are presented and discussed in the paper.

We will focus on synonyms, hyponyms and hiperonyms thanks to which the database of intelligent Chatterbot has been created. Authors also introduce their own algorithm, which could be more efficient than others available on the market in finding and generating words by Chatterbot.

To conclude we are going to compare present software environment and programming languages of intelligent chatterbots with our designed system.

2 Introduction to Botics

Botics [9] is a new branch of science and it is being invented recently. The definition of this term has been not found neither in dictionaries nor encyclopedias. Although some terms, like bot or chatterbot could be found in the literature. We would stress that it is possible to define Botics as the science exploiting the "knowledge of behavior", modeling, and simulation of digital forms. Final product of Botics would be defined as more efficient interactions between humans and computers.

The bots [9] are defined as the computer programs (codes) characterized by freedom of functionality, and acting thanks to solutions from artificial intelligence (neural network systems, expert systems, and AIML). They are commonly used on share market, indexing of stock exchange, for searching the specific information in the Internet, etc. The next important meaning in Botics is Avatar [9]. It could be successfully used for graphic representation of bot in human – computer interaction (communication).

From the point of view of botics development, special attention should be paid to the following subjects: first - virtual assistants (chatterbots, virtual humans, animals as well as the virtual world), second – implementing of the artificial intelligence in computer games (simulation games and bots).

3 History of Chatterbots

The first chatterbot – Eliza [6] was invented in 1966, than developed. Creating such a bot is strictly related to generating the proper answer for defined (asked) question. The author of Eliza, thus the name of the first bot, has decided that the code would answer the question by introducing the question phrase instead basic phrase, i.e.:

The Main sentence: I am happy person today.

The sentence after the modification: Are you talking to me because you are happy person today?

One should notice that this kind of answers generation has many drawbacks, the number of errors is quite significant. There are as follows:

a) the generated answer is always the question,
b) the answer is not logical,
c) inaccuracy,
d) semantic and grammar errors.

We could point out the generated phrase, which is defined as question phrase, is logically and grammatically incorrect. Due to generated errors, it was not possible to accept this algorithm of answering as correct. Therefore it has been decided to implement other methods, which are taken till now for generating answers.

4 Popular Methods of Searches and Generating of Answer

The most important term of the bot is, in our opinion, the text analyzer. The general task of text analyzer is scanning procedure in order to understand what is written. To achieve this goal two browsers have been built: *Matrix browser and Specialized browser*. The special browser [5] compares text written by user with sentences in database. The browsers mentioned above exploits knowledge stored in the form of files in database system. Each file contains data suitable for both browsers. The text analyzer [5] should be equipped with the code (module) for searching the answer. There are two stages of searching procedure.

During the first step special browser starts. If searching procedure is successful (positive answer) the chatbot returns the answer. If not, the second step takes place.

During the second step matrix browser [5] starts. Again, if searching procedure is successful the chatbot returns the answer. If answer is negative the second step of searching is activated, then if searching procedure is successful the chatbot returns the answer.

If answer is negative one-word file is activated. Each searching procedure has its probability of finding proper answer. For the special browser it is even 96% of probability, which means a quite high level. We could stress that for the matrix browser it is approximately 66% of probability.

The lowest probability is for one-word file; we estimate approximately of 15%. One could improve accuracy by adding a huge amount of records to database. Even by enlarging the database it is not possible to achieve of 100% probability by implementing this method.

The general question is how to improve the probability of proper answer?. The answer is so simple - by creating modified browsers. The level of probability would be reached successfully by implementing to the algorithms psycholinguistic grammar, AI algorithms, “communication grammar” [7], etc.

5 AIML and Decision Trees in Chatterbot Creating

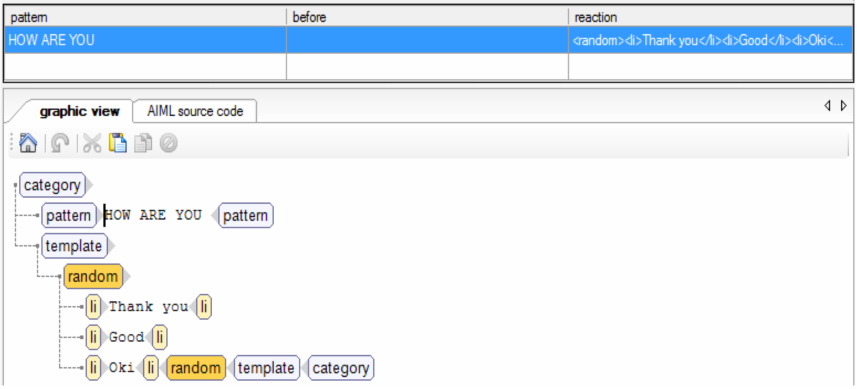
For Chatterbot programming, we could use AIML (Artificial Intelligence Markup Language). AML is similar to XML language, which means that both are based on markers. One of the most popular tools for Chatterbot programming is Gaibot AIML Editor by Springwald Software[4].

This one has got user friendly graphic interface, AIML built in interpreter and an empty knowledge base, which is being created during the process of Chatterbot composing. One could see exemplary Graphic View (see Figure 1), used for Chatterbot knowledge base creation. Software, based on AIML, contains three important functional blocks: the user interface, AIML interpreter, and the AIML knowledge base.

The principle of operation is as follows: User asking Chatterbot a question starts the action between the interface and the knowledge base, which next starts AIML interpreter [4].

The Interpreter's task is to find the string of signs introduced by a user and the string of signs stored in the knowledge base corresponding to introduced string. If the string of signs contained the database is found, the answer is generated automatically. On the contrary, if the string is not found then the answer (phrase) is generated randomly or the message of phrase misunderstanding is displayed.

The knowledge engineers try to improve the code by implementing new functions enabling to generate pseudo- random answers if the string of signs introduced by the user does not exist in the Chatterbot knowledge base. The flowchart of Chatterbot operation, based on AIML by Springwald Software, is presented in Figure 2.



pattern	before	reaction
HOW ARE YOU		<random><li>Thank you</li><li>Good</li><li>Oki<...

Fig. 1. Graphic View – Gaibot AIML Editor

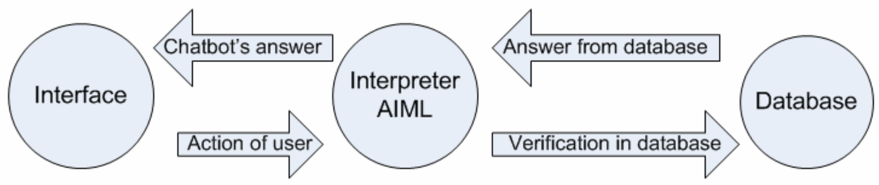


Fig. 2. Flow chart of Chatterbot operation

Another way of Chatterbot creation is the use of decision trees which are present in CLIPS(C Language Integrated Production System) [3] software elaborated by Software Technology Branch (STB), NASA/Lyndon B. Johnson Space Center. The exemplary decision tree (selected part) is presented in Figure 3.

The fault of the software got through the CLIPS programme is its one way concluding procedure. Thus we could get the Chatterbot which is dedicated for answering the direct questions. Such Chatterbot could be used as the help desk in a customer service or in the bus and train information centres.

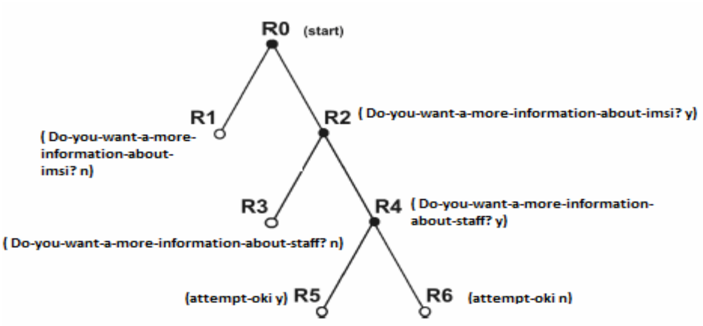
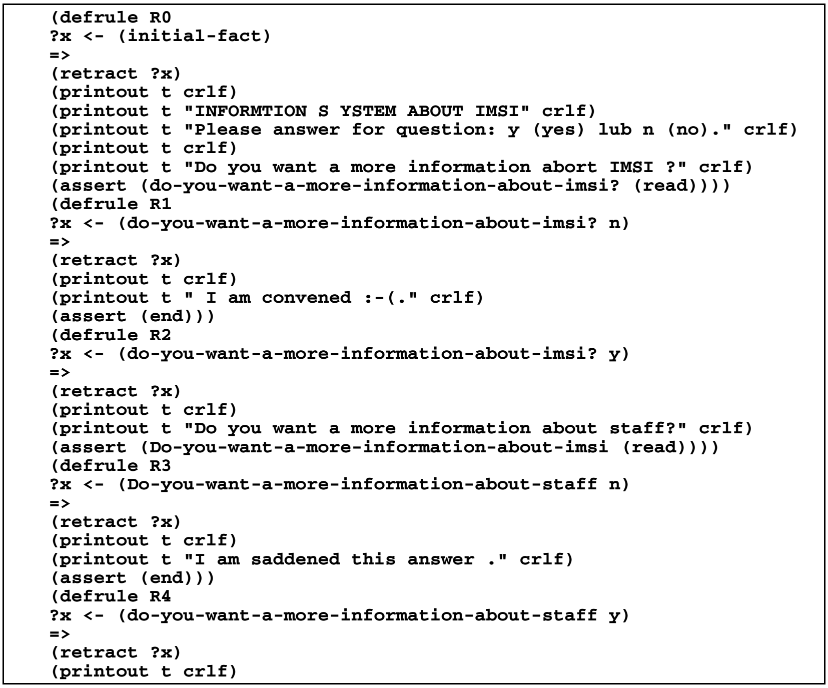

R0 (start)
R1
(Do-you-want-a-more-information-about-imsi? n)
R2 (Do-you-want-a-more-information-about-imsi? y)
R3
(Do-you-want-a-more-information-about-staff? n)
R4 (Do-you-want-a-more-information-about-staff? y)
(attempt-oki y) R5
R6 (attempt-oki n)

Fig. 3. Decision tree based on the CLIPS language

Unfortunately, there is a lack of knowledge and the lack of concluding algorithms. Moreover creation the code is also so complicated, and as well code instability being operated under Windows operating system (see Figure 4).



```
(defrule R0
?x <- (initial-fact)
=>
(retract ?x)
(printout t crlf)
(printout t "INFORMTION S YSTEM ABOUT IMSI" crlf)
(printout t "Please answer for question: y (yes) lub n (no)." crlf)
(printout t crlf)
(printout t "Do you want a more information abort IMSI ?" crlf)
(assert (do-you-want-a-more-information-about-imsi? (read))))
(defrule R1
?x <- (do-you-want-a-more-information-about-imsi? n)
=>
(retract ?x)
(printout t crlf)
(printout t " I am convened :-(." crlf)
(assert (end)))
(defrule R2
?x <- (do-you-want-a-more-information-about-imsi? y)
=>
(retract ?x)
(printout t crlf)
(printout t "Do you want a more information about staff?" crlf)
(assert (Do-you-want-a-more-information-about-imsi (read))))
(defrule R3
?x <- (Do-you-want-a-more-information-about-staff n)
=>
(retract ?x)
(printout t crlf)
(printout t "I am saddened this answer ." crlf)
(assert (end)))
(defrule R4
?x <- (do-you-want-a-more-information-about-staff y)
=>
(retract ?x)
(printout t crlf)
```

Fig. 4. Chatterbot code in the Clips language

Finally, we could stress the above algorithms are not correct, and they generate to many errors. In order to reduce the errors generated by these algorithms it is necessary to create a special database, and algorithms based on psycholinguistics and natural language rules.

6 The Implementing of Psycholinguistics Rules in Chatterbot Creation

Let us point out that Psycholinguistics [1] covers the psychological base of language functioning, which means the language is transformed by humans. The aim of Psycholinguistics is to give the tool facilitating conceptual (not alphabetical) search of database dictionary.

Psycholinguistics enables the creation of language database system based on psycholinguistic theories about human "lexical memory"[2]. That means a large set of semantic dependences between words. Each word with the psycholinguistics use means the connection of semantic concept and the phrase with syntactic role.

One of the most important concept used in the work, mentioned above, is the concept of synonym.

Synonimia [2] *is so narrow meaning, thus if unit a is equivalent to unit b in a particular context, and to unit c in another one (which means that it is impossible to find natural context for b and c). Therefore, it should be assumed that we deal with two units a, while the meaning of one unit is defined by the synset {a,b}and for another one {a,c}.*

Another important notion is the set of synonyms mentioned above. Such a set is called synsets [2]. In the aim of illustrating it, we present the example of arbitrarily selected synset from the Chatterbot database:

Car{ auto, automobile, wheels, set of wheels}
Wheels { car, auto, automobile, set of wheels}

During the Chatterbot data base creation we have also used hyponyms and hypernyms, which being connected with the other database records could give the chance to get better results.

Hyponym [2] is defined as a word whose semantic meaning is more narrow than the basic word. Exemplary, car is the hyponym of vehicle and the hypernym [2] of the word car is the police car (the general word with superior meaning in the respect to

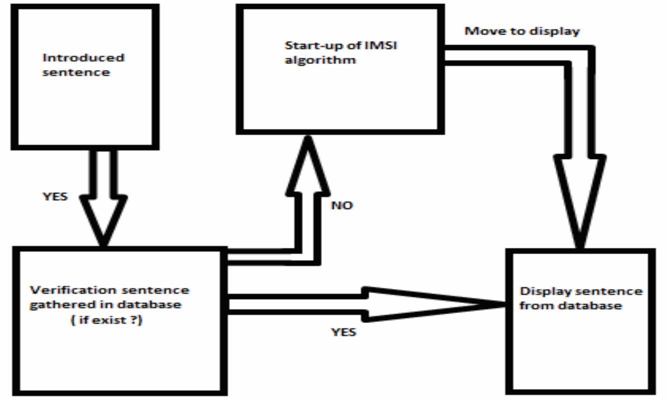

Introduced sentence
Start-up of IMSI algorithm
Move to display
YES
NO
Verification sentence gathered in database (if exist ?)
YES
Display sentence from database

Fig. 5. The algorithm of searching the answer designed for Chatterbot (IMSI algorithm)

other words). The algorithm of searching the answer designed for Chatterbot is presented in Figure 5, and database schema is presented in Figure 6 (selected tables).

The principle of operations of the algorithm (proposed by authors) lays in entering the phrase (sentence) to the code, then checking in if the phrase exists in the database. If result of searching is positive, then the answer is displayed. If result of searching is negative, then our algorithm is initiated. We would point out that database contains input and output phrases, moreover we have also information about the terms of which phrase is composed. If one of the term does not exist in the phrase, matching procedure is activated to substitute synonym, hyponym or hypernym instead term.

We stress that proposed in the paper substituting procedure is so efficient in finding proper answer, even the answer has been not found for the first stage of searching.

Describing our algorithm we select four terms of language and three psycholinguistics rules. As one could see each term of language has its own synonym, hypernym and hyponym.

Finally, main table called "Sentences in and out" is created, in which full sentences and their parts are being stored.

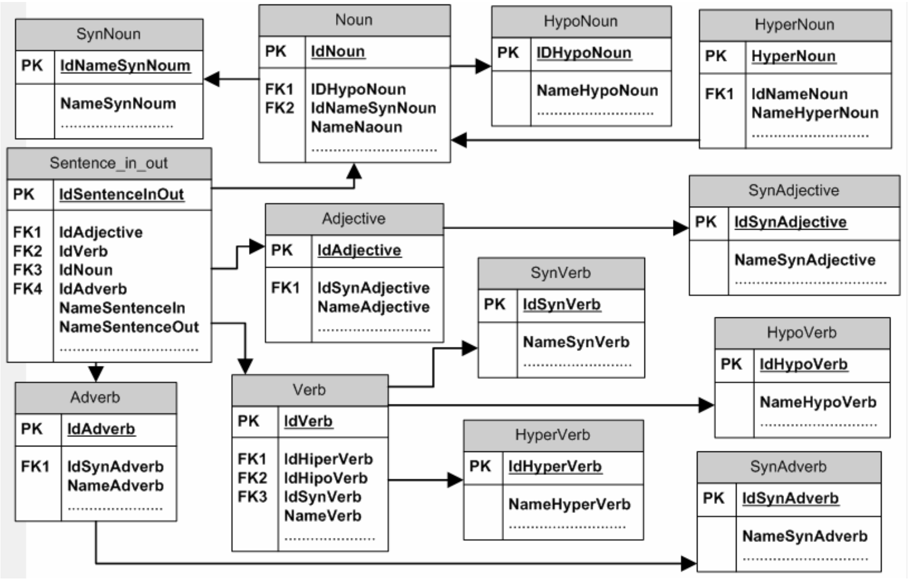


Fig. 6. The database (selected tables)

7 Comparative Analysis of the Algorithms

For the comparative analysis of the algorithms efficiency in searching and generating answers we exploited the following codes:

- Gaibot AIML Editor - Springwald Software [4]
- Authors' algorithm (IMSI algorithm) based on psycholinguistics rules.

The following criteria have been defined to carry out the comparative analysis of algorithms efficiency (see Table 1):

- time of searching the sentence
- the correctness of finding the sentence
- algorithm structure
- the possibility of database extension
- time of sentence generating
- the accuracy of correctly generated answers.

For the tests calculation we defined ten sentences, treated as criteria for comparative analysis, and not connected. It is so important to notice that none of these criteria of searching and generating the answers in Chatterbot is perfect. The errors, which appear, are caused mainly by the substantial misunderstanding of the sentence.

The available algorithms do not carry out the analysis of the content of the sentence (phrase). Instead the above they could compare the sentence with the pattern stored in the database or they could exploit decision tree. It is so important to point out that in the case of the algorithm proposed by authors it could be possible to reduce the errors during searching and generating the answers procedure.

Table 1. The results of comparative analysis

Criteria of comparative analysis	GaiBOT AIML	Authors' Algorithm
time of searching the sentence	fast	Fast
the correctness of finding the sentence	6/10	**8/10**
algorithm code	simple	more advanced
the possibility of data base enlargement	yes	Yes
time of sentence generating	fast	Fast
the accuracy of correctly generated answers	6/10	**8/10**

Concluding there is no algorithm giving of 100% of the probability of correct answer. The use of psycholinguistics principle (rules) and defining the proper relations could reduce the inaccuracy in searching the sentence, and what is more the generated sentences are more correct.

The time of searching and generating the sentences is quite short. In both algorithms there is possibility of data base extension by means of introducing the terms of language or psycholinguistics rules.

I our opinion, it seems to be impossible to create such an algorithm giving the correct answer with 100% probability.

The problem with the creating such a precise software lays in the difficulty to write down the meaning of words by means of the computer code.

References

1. Gleason, J.B., Ratner, N.B.: Psycholingwistyka Gdańskie Wydawnictwo Psychologiczne, Sopot 2005, wyd. 1 (2005) (in Polish), ISBN 83-89574-63-2
2. Derwojedowa, M., Zawisławska, M.: Relacje semantyczne Słowosieci. Biuletyn PTJ LXIII, 2007, str. 217-230 (2007) (in Polish)
3. Popolizio, J.: CLIPS: NASA's COSMIC Shell. Artificial Intelligence Research, August 1 (1988)
4. GaiBot Aiml Editor Standard V1.2 B0 – Springwald Software
5. Kosiorowski, P., Wiak, S.: Intelligent Chatbot's review of technology. In: 2nd Polish and International PD Forum - Conference on Computer Science, Smardzewice, Poland, Conference Proceeding (2006)
6. Kosiorowski, P., Wiak, S.: Generating the chatbot's answer using: special and matrix browsers and markov's algorithm. In: 3rd Polish and International PD Forum - Conference on Computer Science, Smardzewice, Poland, Conference Proceeding (2007)
7. Kosiorowski, P., Wiak, S.: Optimization of answer searching algorithms in chatbot's with the usage of genetic operators and innovative algorithm. In: 4th Polish and International PD Forum - Conference on Computer Science, Smardzewice, Poland, Conference Proceeding (2008)
8. `http://www.codeproject.com/KB/library/ProjectEliza.aspx`
9. `http://www.aibotworld.com/botyka.html`

UMTS Base Station Location Planning with Invasive Weed Optimization

Rafał Zdunek and Tomasz Ignor

Institute of Telecommunications, Teleinformatics and Acoustics,
Wroclaw University of Technology,
Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland

Abstract. The problem of finding optimal locations of base stations, their pilot powers and channel assignments in UMTS mobile networks belongs to a class of NP-hard problems, and hence, metaheuristics optimization algorithms are widely used for this task. Invasive Weed Optimization (IWO) algorithm is relatively novel and succussed in several real-world applications. Our experiments demonstrate that the IWO algorithm outperforms the algorithms such as Evolutionary Strategies (ES) and Genetic Algorithms (GA) for optimizing the UMTS mobile network.

1 Introduction

Planning of UMTS mobile networks involves using static Monte Carlo (MC) simulations to efficiently optimize power resources in the designed network [1,2]. The optimization can be performed with respect to many network parameters such as base stations locations [3,4,5], a number of sectors and antenna configurations [6], their azimuth and tilt angles, their pilot powers as well as channel assignments [7].

In this paper, we attempt to find optimal locations of BSs, their pilot powers as well as channel assignments for one snapshot of MC simulations in a UMTS uplink transmission. We formulate the optimization problem with respect to power resources, where the cost function to be minimized involves the uplink transmission powers which (except for a very simplified model) cannot be explicitly expressed as analytical functions of the parameters to be optimized. To compute the unknown powers, the system of linear equations subject to nonnegativity constraints must be solved, where the coefficients of the system matrix are discontinues and nonlinear functions of BS locations. Thus the optimization problem belongs to a class of NP-hard problems with large number of variables, and hence, the usage of metaheuristics is well justified.

To tackle this problem, several approaches based on different kinds of metaheuristics have been proposed. Amaldi *et al.* [3] estimated BS locations in UMTS using the greedy, reverse greedy randomized and Tabu Search (TS) algorithms. The usage of TS algorithm for planning a cellular network has been also studied by Lee and Kang [8]. Several researchers [9, 10] propose to apply Simulated Annealing (SA) algorithms for locating and configuring BSs. Another

L. Rutkowski et al. (Eds.): ICAISC 2010, Part II, LNAI 6114, pp. 698–705, 2010.
© Springer-Verlag Berlin Heidelberg 2010

metaheuristics-based approach to optimization of UMTS networks concerns the application of Genetic Algorithms (GA) [2,5,11].

In this paper, another bio-inspired metaheuristics, named Invasive Weed Optimization (IWO), is used for locating BSs in UMTS. The IWO algorithm was proposed by Mehrabian and Lucas [12] in 2006, and since then, it has already found many real-world applications such as antenna design and configuration [13], recommender systems [14], piezoelectric actuator positioning [15], and study of electric market dynamics [16]. To our best knowledge, the IWO algorithm has not been applied to any UMTS optimization task yet. Our studies demonstrate that the IWO algorithm in application to positioning BSs in UMTS outperforms the most robust metaheuristics such as evolutionary strategies and GA.

The layout of the paper is as follows. The next section discusses the uplink transmission model in UMTS and formulate the optimization problem. Section 3 describes the IWO algorithm. The experiments are given in Section 4. Finally, the brief conclusions are presented in Section 5.

2 Model

Let us assume that in the UMTS system we have K users assigned to M base stations. The uplink transmission is described by the model $\forall k \in \{1, \ldots, K\}$:

$$\frac{l(\phi_k, k)p_k}{\sum_{j \neq k} l(\phi_k, j)p_j + \eta(\phi_k)} = \gamma_k, \tag{1}$$

where p_k is the unknown power of the signal sent by the k-th Mobile Station (MS), ϕ_k is the index of the BS to which the k-th MS is assigned, $l(\phi_k, j)$ is the link gain between the j-th MS and the ϕ_k-th BS, $\gamma_k = \frac{R_b^{(k)} E^{(k)}}{B}$ is the target Signal-to-Interference Ratio (SIR) of the k-th MS, and $\eta(\phi_k) = (\eta_0 B)^{(k)}$ is the thermal noise of the ϕ_k-th BS. Following then, $R_b^{(k)}$ is the data bit-rate of the k-th MS, $E^{(k)}$ is the required ratio of bit energy to noise and interference power spectral density, B is the channel bandwidth, and η_0 is the spectral density of thermal noise.

The model (1) can be easily expressed in the equivalent matrix form:

$$\boldsymbol{A}\boldsymbol{p} = \bar{\boldsymbol{\eta}}, \tag{2}$$

where $\boldsymbol{p} = [p_1, \ldots, p_k]^T \in \mathbb{R}^K$, $\bar{\boldsymbol{\eta}} = [\eta(\phi_1), \ldots, \eta(\phi_K)]^T \in \mathbb{R}^K$, and $\boldsymbol{A} = [a_{ij}] \in \mathbb{R}^{K \times K}$, where

$$a_{ij} = \begin{cases} -l(\phi_i, j), & \text{for } i \neq j \\ \frac{l(\phi_i, i)}{\gamma_i}, & \text{for } i = j \end{cases} \tag{3}$$

The link gain in (3) can be expressed as $l(\phi_i, j) = g(\phi_i, j)\alpha(\phi_i, j)$, where $g(\phi_i, j)$ is the directional gain for a pair of the ϕ_i-th BS and the j-th MS antennas, and $\alpha(\phi_i, j)$ is the propagation loss between the ϕ_i-th BS and the j-th MS.

Obviously, both functions $g(\phi_i, j)$ and $\alpha(\phi_i, j)$ depend on the location of the MS with respect to the BS. In general, the functions are nonlinear, discontinues, and often given by lookup measurement tables. However, without loss of generality, we assume $g(\phi_i, j) = G_0$, where G_0 is a constant, and in practice this case means that a pair of BS-MS antennas has an omnidirectional horizontal gain. According to the propagation model used in [17], we have $\alpha(\phi_i, j) = 128.38 + 35.2 \log_{10}(d_{ij})$, [dB], where d_{ij} is the distance between the ϕ_i-th BS and the j-th MS.

Since we consider only the uplink transmission, a handoff may be neglected, which means that each MS is assigned to only one BS. A decision on assignments is taken from the Received Signal Code Power (RSCP) measured by a receiver on the Common Pilot CHannel (CPICH). If one or more CPICH signals are detected, such a BS is selected whose the CPICH signal is received with the highest RSCP. A CPICH power affects the assignments and the approximate radius of a cell. Thus, the right setting of this power in each BS is important for optimization of power resources.

Additionally, to better exploit power resources, all the data transmission streams can be divided into several independent channels. We assumed that our analyzed system has 3 data transmission channels (beside CPICH), and each MS can use any accessible data channel. Thus all the network links in our system can be superposed from 3 independent layers of channel links. The channel assignment obviously affects power resources, and this is also one of the optimization tasks.

All the variables to be optimized can be modeled by the unknown matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4] \in \mathbb{R}^{M \times 4}$, where the column vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ contain corresponding x-axis and y-axis coordinates of M BSs, $\boldsymbol{x}_3$ contains the CPICH powers in each BS, and $\boldsymbol{x}_4$ informs on the channel assignment of the uplink transmission links in each BS. Each variable is box constrained. We have the following ranges for the variables $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$: $0 \leq \boldsymbol{x}_1 \leq X_{max}$, $0 \leq \boldsymbol{x}_2 \leq Y_{max}$, where X_{max} and Y_{max} define the size of a rectangular area covered by the UMTS system. For the CPICH power, we assumed 19 [dBm] $\leq \boldsymbol{x}_3 \leq$ 32 [dBm], and $\boldsymbol{x}_4 \in \{1, 2, 3\}$ is a discrete variable containing channel assignments.

Considering the variables to be optimized, the model in (2) can be rewritten as $\boldsymbol{A}(\boldsymbol{X})\boldsymbol{p}(\boldsymbol{X}) = \bar{\boldsymbol{\eta}}$, where each a_{ij} depends on $\boldsymbol{X}$. The optimization problem is defined as follows:

$$\min_{\boldsymbol{X}} \Psi(\boldsymbol{X}), \quad \text{s.t} \quad \boldsymbol{A}(\boldsymbol{X})\boldsymbol{p}(\boldsymbol{X}) = \bar{\boldsymbol{\eta}}, \;\; \boldsymbol{p}(\boldsymbol{X}) \geq 0, \tag{4}$$

and the above box constrains for $\boldsymbol{X}$, with $\Psi(\boldsymbol{X}) = ||\boldsymbol{p}(\boldsymbol{X})||_1$, where $||\boldsymbol{p}||_1$ denotes the l_1-norm of $\boldsymbol{p}$. The system of linear equations in (4) subject to nonnegativity constraints need to be solved in each iteration of the metaheuristics. The fast methods designed for this purpose can be found, e.g. in [18].

3 Invasive Weed Optimization

The IWO algorithm [12], which belongs to a class of metaheuristics global optimization, directly searches through the space of feasible solutions with some evolutionary-based strategy, inspired from a common phenomena in agriculture that is colonization of invasive weeds. A weed is any plant such as tree, vine, shrub, or herb that grows in an undesirable field, and due to its vigorous and invasive habits seriously threatens to the desired and cultivated plants. A colony of weeds tries to improve its fitness to the environment to live longer. Weeds vegetate, flower, and produce seeds that are spread out in the environment by wind, water, animals, etc. When they find good conditions to sprout, they grow, struggle for existence with competitors, produce seeds, and the reproduction repeats. After several periods of reproduction, the population becomes better adapted to the environment, and the individuals (weeds) live longer. This natural ecological system of weed colonization is exploited in the IWO algorithm that somewhat resembles the evolutionary strategies. The IWO algorithm has the following steps:

Algorithm 1. (**IWO**)

```
Initialization  Create initial population $\mathcal{X}$ of $N_0$ individuals (weeds):
                $\mathcal{X} = \left\{ \boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}, \ldots, \boldsymbol{X}^{(N_0)} \right\}$, spread out in
                the whole space of feasible solutions,
For             $t = 1, 2, \ldots, T$ do
  Step 1:       Evaluate $\forall n : \Psi(\boldsymbol{X}^{(n)})$, where $n = 1, \ldots, N_0$,
  Step 2:       Sort the population $\mathcal{X}$ in the ascending order of $\Psi(\boldsymbol{X}^{(n)})$,
  Step 3:       Create the subset $\mathcal{X}_c$ from the first $N_c$ individuals of $\mathcal{X}$,
  Step 4:       Apply a given reproduction scheme:
  Step 4.A.     Each $\boldsymbol{X}^{(c)} \in \mathcal{X}_c$ $(c = 1, \ldots, N_c)$ copy $S^{(c)}$ times, where
                $S^{(c)} \in [S_{min}, S_{max}]$, and put all the copies to $\mathcal{X}_s = \left\{ \boldsymbol{X}^{(s)} \right\}$ ,
  Step 4.B.     Generate seeds over the search space: For $s = 1, \ldots, |\mathcal{X}_s|$ :
                $\boldsymbol{X}^{(s)} \leftarrow \boldsymbol{X}^{(s)} + \Delta^{(s)}$, where $\Delta^{(s)} \sim \mathcal{N}(\mu(\boldsymbol{X}^{(s)}), \sigma_t)$ ,
  Step 5:       Create the set $\mathcal{X}_f = \mathcal{X} \bigcup \mathcal{X}_s$, and sort it as in Step 2,
  Step 6:       Create the population $\mathcal{X}$ from the first $N_{max}$ individuals of $\mathcal{X}$.
End
```

The initial population can be randomly distributed but should be considerably diversified. Let N_0 be the number of individuals in the initial population. The fitness of the individuals is evaluated with the cost function $\Psi(\boldsymbol{X})$ in (4). A given number (N_c) of individuals with best fitness produces seeds according to the seed reproduction scheme. More seeds are produced by the weeds with better fitness. The seeds are then randomly spread out around the parent weeds according to the normal distribution with the mean of the parent weed, and the standard deviation given by

$$\sigma_t = \left(\frac{T-t}{T}\right)^{\nu} (\sigma_{start} - \sigma_{stop}) + \sigma_{stop}, \tag{5}$$

where t denotes the current iteration (generation), T is the maximal number of iterations, (if the stopping criterion is given by the number of iterations), σ_{start} and σ_{stop} are the corresponding initial and final (for the last generation) values of the standard deviation, and ν is a nonlinear modulation index. When t increases, σ_t goes down, which means that the neighborhood around the parent weed from which the offsprings are generated gradually shrinks with iterations. Thus, the algorithm samples the space of feasible solutions denser around the individuals with better fitness. After evaluating fitness of the offsprings, a new population is generated from the parent and offsprings with the best fitness. Other individuals are eliminated.

4 Experiments

Our tests[1], [2] are performed for one random snapshot of the uplink in the WCDMA network with single omnidirectional antennas in BS/MS. We assumed 25 BSs and 250 MSs located on the squared area of 5×5 [km^2]. One third of MSs are randomly distributed according to a screw Gaussian distribution, but the others are uniformly distributed. The black points shown in Fig. 1 (left) denote positions of MSs. Half of MSs works with a voice service ($R_b = 12.2$kbps), and the other half with a data service ($R_b = 64$kbps).

In the IWO algorithm, the number of individuals in the population changes with generations. We started from $N_0 = 1000$, and the maximum size of the populations is set to $N_{max} = 1500$. Similarly as in many evolutionary-based optimization algorithms, the reproduction is strictly related to the fitness of individuals (weeds) in the population. A number of seeds in one generation depends on two parameters such as the number of seeds generated by one individual (parent weed), and the other one is the number of individuals from the base population allowed to make reproduction. We assumed the following seed reproduction scheme: each individual from the first 10 individuals with best fitness generates 10 seeds. Then, each individual from the next 10 parent weeds produces 9 seeds, and the last 10 individuals from the first 100 individuals with best fitness produce one individual each. Totally, we have $N_c = 100$ individuals with best fitness allowed to produce 550 seeds, i.e. $|\mathcal{X}_s| = 550$. In the t-th generation, the seeds are spread out with the rule (5). The initial and final standard deviations are defined as 1/3 and 1/100 of the range of a given variable, respectively.

[1] The experiments are carried out with the algorithms implemented in Matlab 7.0 and tested by T. Ignor in [19].

[2] The calculations have been carried out in Wroclaw Centre for Networking and Supercomputing (http://www.wcss.wroc.pl) with the Nova cluster, Grant No: 127.

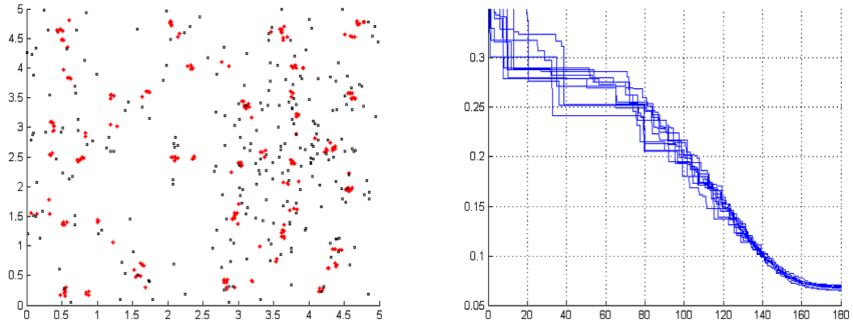


Fig. 1. Results obtained with the IWO for 10 independent initializations: (left) localizations of MSs (black points), estimated localizations of BSs (red points), (right) convergence (fitness) versus the simulation time [in minutes]

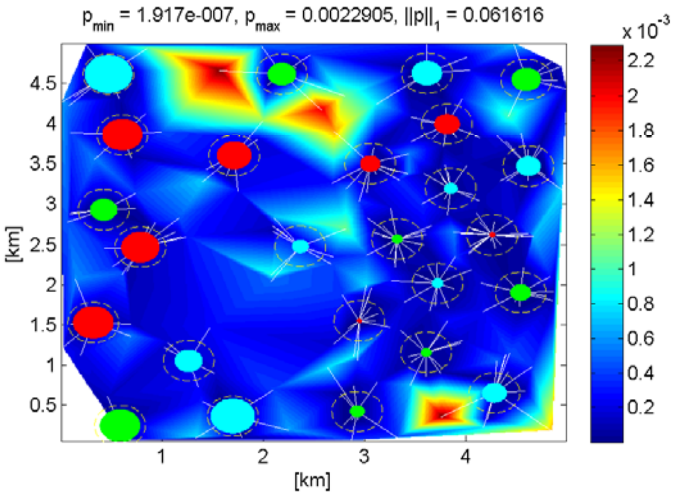


Fig. 2. Multi-plot graphical illustration of the results obtained with the IWO algorithm: positions of BSs (centers of color filled circles), MSs assignments (white solid lines), channel assignments (colors of the circles around BSs), relative CPICH powers (radii of the color filled circles)

Figs. 1–2 present the results obtained with the IWO algorithm applied to the UMTS problem. Fig. 1 (left) illustrates the positions of MSs (black points), and the BS positions (red points) estimated with the IWO algorithm initialized randomly 10 times. Fig. 1 (right) plots the values of the cost function (convergence) of 10 independent initializations versus the running time (max. 3 hours). Note that both figures present the consistent results, and each initialization with the IWO is convergent to nearly the same approximation after about 160 minutes. Fig. 2 illustrates the multi-plot of several variables. The background presents

Table 1. Mean-values and standard deviations (in parenthesis) of the cost function $\Psi(\boldsymbol{X})$ (fitness) averaged over 10 runs, and obtained using various algorithms

Algorithm	ES (1+1)	ES $(\mu + \lambda)$	RW-GA	T-GA	IWO
Fitness	269.5 (32.7)	70.6 (5.1)	75.1 (3.4)	74.8 (3)	67.9 (1.7)

linearly interpolated powers of the signals sent by MSs distributed on the 2D square area (5km × 5km). The assignments of MSs to each BS are plotted by white lines (segments) connecting MSs with their BS. Each BS is located in the center of one color filled circle. There are 3 colors (red, green and cyan) that denote the channel assignment. That is, the BSs with a green filled circle work on the first data channel but the BSs with a cyan one on the second data channel. The radius of the color filled circle informs on the CPICH power with respect to the maximal CPICH power that is denoted by a white dashed circle.

For comparison, we present the results [19] obtained with other metaheuristics such as the evolutionary strategies (1+1) and $(\mu+\lambda)$, and the GA with Roulette Wheel (RW) selection (RW-GA) and Tournament (T) selection (T-GA), where the succession in both GA algorithms is defined by the elitist strategy with the 3-points crossover. For the GA algorithms, we set: $N_0 = 1000$, $p_c = 0.7$ (crossover probability), $p_m = 0.01$ (mutation probability for each bit). In the strategie $(\mu + \lambda)$, the initial base populations for μ and λ account for 400 and 2800 individuals, respectively. All the tested algorithms operated with about 3 hours. The mean-values and standard deviations (in parenthesis) of the cost function (fitness) averaged over 10 runs are given in Table 1.

5 Conclusions

The IWO algorithm gives the approximation with the lowest mean-value of the cost function (the best fitness) and the smallest standard deviation. The approximations for the BS positions are the most consistent. The channel assignments need more research and it is still an open question.

References

[1] Laiho, J., Wacker, A., Novosad, T.: Radio Network Planning and Optimization for UMTS. John Wiley and Sons, Chichester (2002)

[2] Nawrocki, M.J., Dohler, M., Aghvami, A.H. (eds.): Understanding UMTS Radio Network Modelling, Planning and Automated Optimisation: Theory and Practice. John Wiley and Sons, Chichester (2006)

[3] Amaldi, E., Capone, A., Malucelli, F.: Planning UMTS base station location: Optimization models with power control and algorithms. IEEE Transactions on Wireless Communications 2(5), 939–952 (2003)

[4] Mathar, R., Schmeink, M.: Optimal base station positioning and channel assignment for 3G mobile networks by integer programming. Ann. Oper. Res. 107(1-4), 225–236 (2001)

[5] Yang, J., Aydin, M.E., Zhang, J., Maple, C.: UMTS base station location planning: A mathematical model and heuristic optimization algorithms. IEI Communications 1(5), 1007–1014 (2007)

[6] Eisenblatter, A., Geerdes, H.F., Koch, T., Martin, A., Wessaly, R.: UMTS radio network evaluation and optimization beyond snapshots. ZIB-Report 04-15, Zuse Institute Berlin, Takustrasse 7, D-14195 Berlin-Dahlem, Germany (2004), http://www.zib.de/Optimization/Projects/Telecom/Matheon-B4

[7] Siomina, I., Yuan, D.: Optimization of pilot power for load balancing in WCDMA networks. In: Proc. IEEE GLOBECOM 2004, Dallas, Texas, vol. 6, pp. 3872–3876 (2004)

[8] Lee, C.Y., Kang, H.G.: Cell planning with capacity expansion in mobile communications: a tabu search approach. IEEE Transactions on Vehicular Technology 49(5), 1678–1691 (2000)

[9] Kocsis, I., Farkas, L., Nagy, L.: 3G base station positioning using simulated annealing. In: Proc. IEEE PIMRC, Lisbon, Portugal, vol. 1, pp. 330–334 (2002)

[10] Hurley, S.: Planning effective cellular mobile radio networks. IEEE Transactions on Vehicular Technology 51(2), 243–253 (2002)

[11] Choi, Y.S., Kim, K.S., Kim, N.: The displacement of base station in mobile communication with genetic approach. EURASIP J. Wirel. Commun. Netw. 2008, 1–10 (2008)

[12] Mehrabian, A.R., Lucas, C.: A novel numerical optimization algorithm inspired from weed colonization. Ecological Informatics 1(4), 355–366 (2006)

[13] Dadalipour, B., Mallahzadeh, A.R., Davoodi-Rad, Z.: Application of the invasive weed optimization technique for antenna configurations. In: Proc. Loughborough Antennas and Propagation Conf., Loughborough, UK, pp. 425–428 (2008)

[14] Sepehri-Rad, H., Lucas, C.: A recommender system based on invasive weed optimization algorithm. In: Proc. IEEE Congress on Evolutionary Computation, Singapore, pp. 4297–4304 (2007)

[15] Mehrabian, A.R., Yousefi-Koma, A.: Optimal positioning of piezoelectric actuators of smart fin using bio-inspired algorithms. Aerospace Science and Technology 11, 174–182 (2007)

[16] Sahraei-Ardakani, M., Roshanaei, M., Rahimi-Kian, A., Lucas, C.: A study of electricity market dynamics using invasive weed colonization optimization. In: Proc. IEEE Symposium on Computational Intelligence and Games (CIG 2008), Perth, Australia, pp. 276–282 (2008)

[17] Eisenbltter, A., Fugenschuh, A., Fledderus, E.R., Geerdes, H.F., Heideck, B., Junglas, D., Koch, T., Kurner, T., Martin, A.: Mathematical methods for automatic optimization of UMTS radio networks. Technical report, MOMENTUM IST (2003)

[18] Zdunek, R., Nawrocki, M.J.: Improved modeling of highly loaded UMTS network with nonnegative constraints. In: Proc. IEEE PIMRC, Helsinki, Finland (2006)

[19] Ignor, T.: Application of evolutionary algorithms for optimization of UMTS mobile network. M.Sc. thesis (supervised by Dr. R. Zdunek), Wroclaw University of Technology, Poland (2009) (in Polish)

Author Index