Advances in Intelligent Systems and Computing 342

Antoni Wiliński
Imed El Fray
Jerzy Pejaś *Editors*

Soft Computing in Computer and Information Science

Springer

Advances in Intelligent Systems and Computing

Volume 342

Series editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland
e-mail: kacprzyk@ibspan.waw.pl

About this Series

The series “Advances in Intelligent Systems and Computing” contains publications on theory, applications, and design methods of Intelligent Systems and Intelligent Computing. Virtually all disciplines such as engineering, natural sciences, computer and information science, ICT, economics, business, e-commerce, environment, healthcare, life science are covered. The list of topics spans all the areas of modern intelligent systems and computing.

The publications within “Advances in Intelligent Systems and Computing” are primarily textbooks and proceedings of important conferences, symposia and congresses. They cover significant recent developments in the field, both of a foundational and applicable character. An important characteristic feature of the series is the short publication time and world-wide distribution. This permits a rapid and broad dissemination of research results.

Advisory Board

Chairman

Nikhil R. Pal, Indian Statistical Institute, Kolkata, India
e-mail: nikhil@isical.ac.in

Members

Rafael Bello, Universidad Central “Marta Abreu” de Las Villas, Santa Clara, Cuba
e-mail: rbellop@uclv.edu.cu

Emilio S. Corchado, University of Salamanca, Salamanca, Spain
e-mail: escorchado@usal.es

Hani Hagras, University of Essex, Colchester, UK
e-mail: hani@essex.ac.uk

László T. Kóczy, Széchenyi István University, Győr, Hungary
e-mail: koczy@sze.hu

Vladik Kreinovich, University of Texas at El Paso, El Paso, USA
e-mail: vladik@utep.edu

Chin-Teng Lin, National Chiao Tung University, Hsinchu, Taiwan
e-mail: ctlin@mail.nctu.edu.tw

Jie Lu, University of Technology, Sydney, Australia
e-mail: Jie.Lu@uts.edu.au

Patricia Melin, Tijuana Institute of Technology, Tijuana, Mexico
e-mail: epmelin@hafsamx.org

Nadia Nedjah, State University of Rio de Janeiro, Rio de Janeiro, Brazil
e-mail: nadia@eng.uerj.br

Ngoc Thanh Nguyen, Wroclaw University of Technology, Wroclaw, Poland
e-mail: Ngoc-Thanh.Nguyen@pwr.edu.pl

Jun Wang, The Chinese University of Hong Kong, Shatin, Hong Kong
e-mail: jwang@mae.cuhk.edu.hk

More information about this series at http://www.springer.com/series/11156

Antoni Wiliński · Imed El Fray
Jerzy Pejaś
Editors

Soft Computing in Computer and Information Science

Springer

Editors

Antoni Wiliński
Faculty of Computer Science and
Information Technology
West Pomeranian University of Technology
Szczecin
Poland

Jerzy Pejaś
Faculty of Computer Science and
Information Technology
West Pomeranian University of Technology
Szczecin
Poland

Imed El Fray
Faculty of Computer Science and
Information Technology
West Pomeranian University of Technology
Szczecin
Poland

ISSN 2194-5357 ISSN 2194-5365 (electronic)
Advances in Intelligent Systems and Computing
ISBN 978-3-319-15146-5 ISBN 978-3-319-15147-2 (eBook)
DOI 10.1007/978-3-319-15147-2

Library of Congress Control Number: 2015930026

Springer Cham Heidelberg New York Dordrecht London

© Springer International Publishing Switzerland 2015
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made.

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Preface

Dear Readers,
We introduce you to a series of carefully selected and reviewed, by independent scientists, papers presented during the 19th Advanced Computer Systems conference—the most significant event for our Faculty of Computer Science and Information Technology. As the Dean I have had the exceptional pleasure to open the ACS conference and as an editor of this book I have had the opportunity to observe the activity and accuracy of young scientists as well as those experienced. All of them struggled for the right to be published in this particular book.

It was the nineteenth edition of the ACS conference, the oldest event in history of our slightly young faculty. The Advanced Computer Systems conference from the beginning of its existence was concentrated on methods and algorithms of artificial intelligence. Further into the future brought new areas of interest concerning technical informatics related to soft computing and some more technological aspects of computer science such as multimedia and computer graphics, software engineering, Web systems, information security, and safety or project management.

The ACS conference has a strong international nature now as well as when Poland was not a member of the European Union. Through years of activity we created strong and satisfying friendships with scientists whose academic achievements made them mentor figures for young scientific talents. I would like to mention, some of the most significant for us, professors who attended the ACS conference from the beginning. They are: Profs. Gisella Facchinetti from Italy, Elisabeth Rakus-Andersson from Sweden, Larysa Globa from Ukraine, Akira Imada from Belarus, Shin-Ya Kobayashi from Japan, Alexandr Dorogov from Russia, Roger French from USA, Olli-Pekka Hilmola from Finland, Kurosh Madani from France, Prof. Alexander Schill from Germany, and Ivan Lopez Arevalo from Mexico. There are obviously more excellent names from the world of science but it is not possible to mention all of them in this Preface.

One of the newest ideas treated, like some kind of social experiment, was the session with the Russian language. This was an idea that helped us to open our conference to Eastern Europe and send an explicit signal that we are ready and open for cooperation in the area of science despite existing international conflicts. We found that experiment very successful because we had the pleasure to receive a few guests from Russia, Ukraine, and Belarus. Most of them declared their presentation also in English but we decided that they should participate in the Russian language session.

This year's event abounded with interesting discussion inspired and provoked by our exquisite guests. Those dynamic scientific debates, very often continued in the lobby, caused the necessity of additional selection of papers finally published in this book.

I am strongly convinced that the intellectual skirmish during the Advanced Computer Systems conference significantly improved the scientific level of presented papers, right after withdrawing some less successful and addition of the better ones.

So I congratulate the conference organizers for its excellent spirit and high scientific level and I hope that the readers will find in this book very interesting source of knowledge and maybe even some motivation for further research.

Szczecin, Poland Antoni Wiliński

Message from Keynote Speaker

It was in 2004 when I attended this conference for the first time. It was held in Elk, Poland. I had coincidentally found the call for papers of this conference on the Internet. I wrote a paper on something like “fitness landscape in a GA search space,” and submitted it. Despite my first participation in this conference, all the participants welcomed me in a very friendly manner. I was glad because many were interested in this topic and we discussed about the topic during the coffee breaks. I was impressed with this academic attitude of the participants as well as the wonderful atmosphere of the conference. I was impressed with the beautiful small town of Elk too.

Then I participated in 2005 also in Elk, with a paper whose title was somewhat of a challenging one, i.e., “When a family of iris flower is normal, then are other families abnormal?” Again I was glad because this attracted the audience’s interest, and I got a lot of academic friends afterwards, mainly in Poland, but not only in Poland but also in Germany, Ukraine, Italy, US., etc.

Then the conference site moved to Miedzyzdroje. I participated in 2006, 2007, 2012, 2013 and this year, 2014. Every time I got something new—stimulating new topic, a new innovative approach to traditional problems, attractive new ways of presentation, new proposal of cooperation, new academic friendships, and so on. The same holds for this year’s conference. The conference web page reads this as the 19th conference of the same name. So I am feeling very happy as I have taken part in this conference more than one-third of the time.

Akira Imada
Professor
Department of Intelligent Information Technology
Brest State Technical University
Belaus
Germany

Organization

Advanced Computer System 2014 (ACS 2014) was organized by the West Pomeranian University of Technology in Szczecin, Faculty of Computer Science and Information Technology (Poland), in cooperation with AGH University of Science and Technology, Faculty of Physics and Applied Computer Science (Poland), Ehime University (Japan) and Polish Academy of Sciences IPI PAN (Poland).

Program Committee Chairs

Włodzimierz Bielecki, West Pomeranian University of Technology in Szczecin, Poland
Andrzej Piegat, West Pomeranian University of Technology in Szczecin, Poland
Jacek Pomykała, Warsaw University, Poland
Khalid Saeed, AGH University of Science and Technology, Poland

Program Committee Co-Chairs

Jerzy Pejaś, Chair, West Pomeranian University of Technology in Szczecin, Poland
Imed El Fray, West Pomeranian University of Technology in Szczecin, Poland
Tomasz Hyla, West Pomeranian University of Technology in Szczecin, Poland

Program Committee Secretaries

Witold Maćków, West Pomeranian University of Technology in Szczecin, Poland
Dariusz Burak, West Pomeranian University of Technology in Szczecin, Poland
Sylwia Hardej, West Pomeranian University of Technology in Szczecin, Poland
Mikhailo, Fedorov, West Pomeranian University of Technology in Szczecin, Poland

International Programming Committee

Artificial Intelligence

Andrzej Piegat, Chair, West Pomeranian University of Technology, Poland
Anna Bartkowiak, Wroclaw University, Poland
Krzysztof Ciesielski, Polish Academy of Sciences, Poland
Gisella Facchinetti, University of Modena and Reggio Emilia, Italy
Akira Imada, Brest State Technical University, Belarus
Janusz Kacprzyk, Systems Research Institute, Polish Academy of Sciences, Poland
Piotr Andrzej Kowalski, Cracow University of Technology, Poland
Jonathan Lawry, University of Bristol, UK
Witold Pedrycz, University of Alberta, Canada
Elisabeth Rakus-Andersson, Blekinge Institute of Technology, School of Engineering, Sweden
Izebela Rejer, West Pomeranian University of Technology, Poland
Leszek Rutkowski, Czestochowa University of Technology, Poland
Zenon Sosnowski, University of Finance and Management in Białystok, Poland
Sławomir Wierzchoń, Institute of Computer Science, Polish Academy of Sciences, Poland
Antoni Wiliński, West Pomeranian University of Technology, Poland
Toru Yamaguchi, Tokyo Metropolitan University, Japan
Jan Węglarz, Poznan University of Technology, Poland

Software Engineering

Włodzimierz Bielecki, Chair, West Pomeranian University of Technology, Poland
Leon Bobrowski, Bialystok Technical University, Poland
Larisa Globa, National Technical University of Ukraine, Ukraine
Janusz Górski, Technical University of Gdansk, Poland
Dietbert Güntter, Technical University of Dresden, Germany
Andriy Luntovskyy, BA Dresden University of Coop. Education, Germany
Andrzej Niesler, Wroclaw University of Economics, Poland
Marcin Paprzycki, Systems Research Institute, Polish Academy of Sciences, Poland
Valery Rogoza, West Pomeranian University of Technology, Poland
Alexander Schill, Dresden University of Technology, Germany

Information Technology Security

Jacek Pomykała, Chair, Warsaw University, Poland
Johannes Bloemer, Paderborn University, Germany
Krzysztof Chmiel, Poznan University of Technology, Poland
Nicolas Tadeusz Courtois, CP8 Crypto Lab, SchlumbergerSema, France
Jos Dumortier, K.U.Leuven University, Belgium
Oleg Fińko, Kuban State University of Technology, Russia
Jerzy August Gawinecki, Military University of Technology, Poland
Zbigniew Adam Kotulski, Polish Academy of Sciences, Poland
Mieczysław Kula, University of Silesia, Poland

Eugeniusz Kuriata, University of Zielona Gora, Poland
Matthias Krause, Mannheim University, Germany
Javier Lopez, University of Malaga, Spain
Arkadiusz Orłowski, Warsaw University of Life Sciences SGGW, Poland
Josef Pieprzyk, Macquarie University, Australia
Vincent Rijmen, Graz University of Technology, Austria
Marian Srebrny, Institute of Computer Science, Polish Academy of Sciences, Poland
Janusz Stokłosa, Poznan University of Technology, Poland

Multimedia Systems

Khalid Saeed, Chair, AGH University of Science and Technology, Poland
Andrzej Cader, Academy of Humanities and Economics in Lodz, Poland
Ryszard S. Choraś, University of Technology and Life Sciences, Poland
Bernard Dumont, European Commission, Information Society and Media Directorate General
Dariusz Frejlichowski, West Pomeranian University of Technology, Poland
Michelle Joab, LIRMM, Universit Montpellier 2, France
Andrzej Kasiński, Poznan University of Technology, Poland
Georgy Kukharev, West Pomeranian University of Technology, Poland
Albert Sangrá, Universitat Oberta de Catalunya, Spain
Władysław Skarbek, Warsaw University of Technology, Poland
Ryszard Tadeusiewicz, AGH University of Science and Technology, Poland
Aleksandr Cariow, West Pomeranian University of Technology, Poland

Design of Information Systems

Shinya Kobayashi, Chair, Ehime University, Japan
Costin Badica, Craiova, Romania
Zbigniew Banaszak, Warsaw University of Technology, Poland
Grzegorz Bocewicz, Koszalin University of Technology, Poland
Anna Burduk, Wroclaw University of Technology, Poland
Albert Dipanda, Le Centre National de la Recherche Scientifique, France
Kathy Horadam, RMIT, Australia
Jarosław Jankowski, West Pomeranian University of Technology, Poland
Jason T.J. Jung, Yeungnam University, Korea
Przemysław Korytkowski, West Pomeranian University of Technology, Poland
Bartłomiej Małachowski, West Pomeranian University of Technology, Poland
Paweł, Pawlewski, Poznañ University of Technology, Poland
Kurt Sandkuhl, University of Rostock, Germany
Bożena Śmiałkowska, West Pomeranian University of Technology, Poland

Referees

Akira Imada
Anna Bartkowiak
Włodzimierz Bielecki
Bobrowski Leon
Robert Burduk
Aleksandr Cariow
Krzysztof Ciesielski
Krzysztof Chmiel
Piotr Czapiewski
Gisella Facchinetti
Mykhaylo Fedorov
Oleg Fińko
Pawel Forczmański
Imed El Fray
Dariusz Frejlichowski
Jerzy August Gawinecki
Larysa Globa
Tomasz Hyla
Sławomir Jaszczak
Mariusz Kapruziak
Shin-ya Kobayashi
Przemysław Korytkowski
Tetiana Kot
Zbigniew Adam Kotulski
Piotr Andrzej Kowalski
Mirosław Kurkowski
Andriy Luntovskyy
Andrzej Niesler
Adam Nowosielski
Arkadiusz Orłowski
Jerzy Pejaś
Andrzej Piegat
Marcin Pluciński
Elisabeth Rakus-Andersson
Izabela Rejer
Walery Rogoza
Leonard Rozenberg
Khalid Saeed
Władysław Skarbek
Zenon A. Sosnowski
Marian Srebrny
Janusz Stokłosa
Ryszard Tadeusiewicz
Sławomir Wierzchoń
Antoni Wiliński
Wojciech Zabierowski
Elena Zaitseva

Contents

Part I Artificial Intelligence

What Is Machine Intelligence? A Case Study: Stock Market Forecasting Agents 3
Akira Imada

Boosted Classifiers for Antitank Mine Detection in C-Scans from Ground-Penetrating Radar 11
Przemysław Klęsk, Mariusz Kapruziak and Bogdan Olech

Detection Accuracy of the Temporary State of Complex Signals Using Phase-Frequency Tracking Methods with Equilibrium and Non-equilibrium Processing 27
Alexander Kochegurov, Elena Kochegurova and Natalya Kupina

Application of Mini-Models to the Interval Information Granules Processing 37
Marcin Pluciński

A Parametric Interval Approximation of Fuzzy Numbers 49
Luca Anzilli and Gisella Facchinetti

Application of the Horizontal Membership Function to the Uncertain Displacement Calculation of a Composite Massless Rod Under a Tensile Load 63
Karina Tomaszewska and Andrzej Piegat

Multi-agent Processes Analysis System in Prediction Task 73
Michał Zabłocki

Method of Static Classifiers Selection Using the Weights of Base Classifiers 85
Robert Burduk

E-book Reader Application Driven by Commands Provided by a Brain-Computer Interface Based on Motor Imagery Paradigm 95
Izabela Rejer and Andrzej Klimek

Comparison Between Mini-models Based on Multidimensional Polytopes and *k*-nearest Neighbor Method: Case Study of 4D and 5D Problems 107
Marcin Pietrzykowski

Part II Design of Information and Multimedia Systems

Sushi: A Lightweight Distributed Image Storage System for Mobile and Web Services 121
Kamoliddin Mavlonov, Yoshinobu Higami and Shin-ya Kobayashi

Verification of Identity Based on Palm Vein and Palm-Print 139
Mariusz Kubanek and Dorota Smorawa

The Concept of Automation in Conventional Systems Creation Applied to the Preliminary Aircraft Design 147
Nikolay Borgest, Maksim Korovin, Andrey Gromov and Aleksey Gromov

Video Stream Analysis for Fish Detection and Classification 157
Paweł Forczmański, Adam Nowosielski and Paweł Marczeski

The Jitter Use to Reduce EMI on the Power Lines for Multi-core Processors 171
Tomasz Król, Leonard Rozenberg and Michał Twardochleb

A Study on Various Dimensionality Reduction Techniques Applied in the General Shape Analysis 181
Katarzyna Gościewska and Dariusz Frejlichowski

Knowledge-Based Approach to COTS Software Selection Processes . . . 191
Agnieszka Konys

The Investment Strategy Based on the Difference of Moving Averages with Parameters Adapted by Machine Learning 207
Antoni Wiliński and Michał Zabłocki

Pattern Recognition in the Japanese Candlesticks 227
Leszek Chmielewski, Maciej Janowicz, Joanna Kaleta and Arkadiusz Orłowski

Part III Information Technology Security

ProToc—An Universal Language for Security Protocols Specifications 237
A. Grosser, M. Kurkowski, J. Piątkowski and S. Szymoniak

Parallelization of Block Encryption Algorithm Based on Piecewise Nonlinear Map 249
Dariusz Burak

Statistical Properties Analysis of Key Schedule Modification in Block Cipher PP-1 257
Michał Apolinarski

Fault Detection in PP-2 Symmetric Block Cipher 269
Ewa Idzikowska

Secure Pseudo-Random Linear Binary Sequences Generators Based on Arithmetic Polynoms 279
Oleg Finko and Sergey Dichenko

Part IV Software Technologies

Energy-Efficient Network Services as Smart Grid Issue 293
Andriy Luntovskyy, Josef Spillner and Volodymyr Vasyutynskyy

Perfectly Nested Loop Tiling Transformations Based on the Transitive Closure of the Program Dependence Graph 309
Wlodzimierz Bielecki and Marek Palkowski

Parallel Byzantine Fault Tolerance 321
Maciej Zbierski

Increasing Web Services Discovery Relevancy in the Multi-ontological Environment 335
Larysa Globa, Mykhailo Kovalskyi and Oleksandr Stryzhak

Advanced Approach to Web Service Composition 345
D. Pukhkaiev, O. Oleksenko, T. Kot, L. Globa and A. Schill

Based on Force-Directed Algorithms Method for Metagraph Visualization 359
Larysa Globa, Maksym Ternovoy, Olena Shtogrina and Oleksandra Kryvenko

Structural Analysis of Singularly Perturbed State Models 371
Walery Rogoza

The Formal Model of Data Mining Algorithms for Parallelize Algorithms 385
Ivan Kholod, Zaynidin Karshiyev and Andrey Shorov

Toward Generalization of Mutant Clustering Results in Mutation Testing 395
Anna Derezińska

TRACO Parallelizing Compiler 409
Marek Palkowski and Wlodzimierz Bielecki

On the Multiplication of Biquaternions 423
Aleksandr Cariow and Galina Cariowa

Model of Collaborative Data Exchange for Inland Mobile Navigation 435
Tomasz Hyla, Natalia Wawrzyniak and Witold Kazimierski

Author Index 445

Part I
Artificial Intelligence

What Is Machine Intelligence? A Case Study: Stock Market Forecasting Agents

Akira Imada

Abstract This chapter is a summary of my talk given during the 19th International Multiconference on Advanced Computer Systems held in Miedzyzdroje, Poland in October 2014. The talk started with the Legg and Hutter's formal definition of what they call Universal Machine Intelligence which aims to measure intelligence of almighty robot, where agents are given an infinitely large number of different types of task to measure its universal intelligence. We take it a consideration what if we apply the formula to only one specific task. We claim the measurement of performance for each of those tasks given in the Legg and Hutter's definition is not sufficient to represent agent's intelligence. Then, we present our ongoing definition of machine intelligence for a specific one task such as forecasting stock market price.

1 Introduction

Two years ago in this conference, I talked about formal definitions of machine intelligence where we discussed whether the Deep Blue who beat Kasparov in 1997 is intelligent? Or, how about Watson who outweighed the ex-human champions in Jeopardy in 2012? [8]. Are they really intelligent? Kasparov later said, "Well, at least it didn't enjoy beating me." Rutter, the ex-human champion of Jeopardy, who was defeated by Watson, jokingly commented, "When Watson's progeny comes back to kill me from the future, I have my escape route planned just in case."

Then degree to how they are intelligent like human? Gregory [1, 22] tried to answer this question using complexity theory. Now we have a fair amount of such definitions. Among others, the definition by [18] is frequently cited to those related papers.

A. Imada (✉)
Department of Intelligent Information Technology, Brest State Technical University,
Moskowskaja 267, 224017 Brest, Belarus
e-mail: akira-i@brest-state-tech-univ.org

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_1

2 Legg and Hutter's Formal Definition of Universal Machine Intelligence

They started with the informal definition that "Intelligence is an agent's ability to achieve goals in a wide range of environments." In their definition, agent interacts with environment. That is, agent observes the environment, makes an action in the environment, and gets rewards from the environment, which yields a history of observation o_i, action a_i, and reward r_i as a series of events. For example, an agent who invests in a stock market observes the market, makes an action, i.e., to buy or to sell stocks, and gets a reward, i.e., profit or loss. This is the reason I chose the stock market forecasting agent as a case study in this talk.

Then agent is defined as a function that takes the current history as input and produces a next action as output. They call this function π.

$$\pi(a_k|o_1r_1a_1o_2r_2a_2\cdots o_{k-1}r_{k-1}). \tag{1}$$

Or it might be a probability function for nondeterministic evaluation. Similarly, environment is defined as a function that produces current observation and reward given the history. They call this function μ.

$$\mu(o_kr_k|o_1r_1a_1o_2r_2a_2\cdots o_{k-1}r_{k-1}a_{k-1}). \tag{2}$$

Or a probability function to be deterministic. Now we can calculate expected value of sum of rewards of agent π for the task μ as

$$V_\mu^\pi = E(\sum_{i=1}^{\infty} r_i). \tag{3}$$

Then intelligence of the agent π is defined as a weighted sum of this expected value of sum of rewards.

$$\gamma(\pi) = \sum_{\mu\in E} w_\mu V_\mu^\pi. \tag{4}$$

The question then is, how will those weights be specified? For the purpose, task μ is translated into a binary string x by the Universal Turing Machine so that we can calculate Kolmogorov complexity K of x. In other words, the length of the shortest program that computes x.

$$K(x) = \min_p\{l(p)|U(p) = x\}. \tag{5}$$

Then the weight for the task μ is specified as

$$w_\mu = 2^{-K(\mu)}, \tag{6}$$

which implies that the smaller the complexity the larger the weight. Thus, Legg and Hutter proposed the equation

$$\gamma(\pi) = \sum_{\mu \in E} 2^{-K(\mu)} V_\mu^\pi \tag{7}$$

as the formal definition of machine intelligence.

Let us recall now the informal definition they started with: "Intelligence is an agent's ability to achieve goals in a wide range of environments." In Eq. (7), V_μ^π is the goal achieved by agent π for task μ. After each of these achieved goals is factored according to the complexity of the task, they are summed up over an infinitely large number of different tasks in order to be *in a wide range of environments*. Legg and Hutter called it *Universal Machine Intelligence*.

3 Machine Intelligence for a Specific Task

Our purpose is, however, to measure intelligence of proposed machine in proposed environment, not to measure a universal almighty robot. To be more specific, we want to compare two different agents reported in two different papers, each proudly claims its magnificent intelligence.

Let Us Be Specific Not Universal. In a daily conversation, we might say, "She is an intelligent dancer," while we know she is not good at Mathematics. Or "This conductor always makes an intelligent interpretation of symphony, but very bad at football." In fact, we do not think Einstein played football intelligently. Hence, a robot could be an intelligent football player, even if it has no sense of philosophy, for example, which we do not care. Intelligence does not need to be universal. Yet as another example, a cooking robot could be said to be intelligent.

From this point of view, let us revisit the Legg and Hutter's definition. As we already saw, their universal machine intelligence is given by

$$\gamma(\pi) = \sum_{\mu \in E} 2^{-K(\mu)} V_\mu^\pi .$$

However, if our concern is a specific one task, the equation will be simply an expected sum of rewards:

$$\gamma(\pi) = V^\pi = E(\sum_{i=1}^{\infty} r_i), \tag{8}$$

which might not appeal us any more. Because intelligence does not always mean highest rewards. Human intelligence is rather spontaneous, flexible, and/or unpredictable, more or less. Or even erroneous sometimes.

Douglas Hofstadter, a cognitive scientist at Indiana University and the Pulitzer prize-winning author of the book Gödel, Escher, Bach: An Eternal Golden Braid [6] said, “How do you get a rigid machine to do very fluid things? That’s a beautiful paradox and very exciting, philosophically,” [7].

Let Machine Intelligence Be More Unpredictable. Intelligence avoids a similar behavior. Assume we live in a foreign country where we are not so conversant in their language, we might frequently have to ask “I beg your pardon?” Then intelligent people try a different explanation for an easier understanding, while others, probably not so intelligent, just repeat the same expression, maybe with a louder voice.

Stock Market Forecasting Agent as a Case Study. One advantage of choosing stock market as a case study is its easy accessibility, not only for professional investors but also for ordinary people. Dow Jones, NASDAQ, S&P 500, Nikkei, …whichever it may be, we can get the daily prices of all the stocks in the market for more than a decade.

Or if we want to make a simulation well controlled, then we may use an artificial stock market such as Santa Fe Artificial Stock market [3, 9, 17]. In fact, Zhang [25] modeled the influence of social networks, like Twitter, in an artificial stock market.

The other advantage is, stock markets are emotional not logical. Its nonlinearity, uncertainty, and dynamic data over time give a clear distinction between intelligence versus efficiency and effectiveness.

Are They Really Intelligent Like Human? Lots of intelligent machine agents who forecast stock prices have already been proposed using machine learning such as Artificial Neural Networks, Hidden Markov Models, Bayesian Belief Networks, Evolutionary Algorithms, Classifier Systems, Fuzzy Sets, and so on…Then how intelligently an agent forecast stock prices?

Or, Not Intelligent at All? Burton Malkiel, 2002 Nobel Laureate in Economics, wrote, “A monkey throwing darts at the WSJ to select a portfolio might be better than the one carefully selected by experts,” in his book A random walk down Wall Street [19].

Is Stock Market Random or Not? Hasanhodzic et al. [5] made an experiment by a Turing-like test to see if human interrogator tells a difference between actual and randomized financial returns, and observed the human interrogator consistently distinguished between the two. Thus the authors refuted a belief that financial markets look random.

Intelligent Agents Should Learn on the Fly! Many proposals use neural networks. However, we have to notice that fixed weights after learning never learn on the fly. Synaptic plasticity during a run (See, e.g., [4]) is necessary at least. Yixian et al.

[24] approached to the stock market forecasting by using dynamic recurrent neural network, though it was still not so matured.

Let Us Require Simple Behavior! Explanations of intelligent people are always simple, while others are not so. This is clear when we think of the Occam's Razor Principle.

Intelligence Must Look Sophisticated as Well. It is a dilemma, but intelligent people also look sophisticated more or less, as once Kluger [11] wrote, "Intelligent individuals are more difficult to learn to know," due to their being sophisticated. We assume so does a machine intelligence.

4 Our Ongoing Definition

Thus far we have incorporated the following factors to define a machine intelligence for a specific task. (i) Behavior must be unpredictable. That is, different action even in a similar situation. (ii) Solution must be one of the simplest. (iii) Better to look sophisticated more or less. (iv) Excellent capability of keeping learning.

To do so we repeat a run in a same condition to see, how similar are results from run to run; how simple is result of each run; how sophisticated each of results looks; and how it learns on the fly; in addition to expected sum of rewards.

Therefore, the intelligence of an agent π for a task μ might be described with the form:

$$V_{\mu}^{\pi} = \sum_{j=1}^{n}\sum_{i=1}^{m} F(a_{ij}) \cdot G(a_{ij}) \cdot H(a_{ij}) \cdot L(a_{ij}) \cdot r_{ij}, \tag{9}$$

where a_{ij} is the ith action in the jth run, r_{ij} is the ith reward in the jth run, and m is a total number of actions in a run, and n is a number of runs repeated in a same environment.

We have not specified each of these functions yet, but as for similarity, for example, as it is the smaller the better, a monotonic decreasing function such as

$$F(a_{ij}) = 1 - (a_{ij}/\sigma)^{\alpha} \tag{10}$$

will be a candidate, where we can control its curvature by α. As for a degree of sophistication, we might use entropy measure:

$$H(a_{ij}) = -\sum p(a_{ij}) \log_2 p(a_{ij}). \tag{11}$$

5 Are They Enough?

What else are necessary for a stock price forecasting agent to be intelligent?

Information from Newspapers is used by almost all human investors. So a stock market forecasting agent needs natural language processing. Kobos et al. [12] approaches from this aspect, though research has not been so matured yet.

As for human language processing, Watson has superb capability, in addition to its hypothesis generation and evaluation. Watson has super learning capability too. It is said that Watson has read 200 million Wikipedia pages. In fact, Citigroup Inc. has been collaborating with IBM since March 2012.

Then would Watson's forecasting stock prices be intelligent? What if we ask Watson, "Which stock is most likely profitable at this very moment?"

Does Watson really understand? Ray Kurzweil, director of engineering at Google, told "…but it doesn't understand that 'if John sold his red Volvo to Mary that involves a transaction or possession and ownership being transferred," [14]. Then what if we ask Watson, "I have a possession of SONY's stocks but not ownership. Should it be transferred for a transaction?"

Emergence of Spontaneous Innovation. Intelligent human investors sometimes devise innovative strategies spontaneously. For instance, we might measure correlations all possible pair of stocks in the market. (See, e.g., [10, 15, 21]).

Then, an innovative strategy might be as follows. (i) Buy a pair of two stocks with a largest negative correlation. Assume they are A and B. (ii) When price of either of these two stocks, assume A, becomes high enough, sell A and buy B, with the money, that might now be cheap. (iii) Wait until the price of B sores, then sell B and buy A whose price now most likely plunges. (iv) Repeat (ii) and (iii). Hopefully a multiple of such a pair.

Long Eye Sights. Intelligent human sometimes neglect a loss just in front, but instead think of total gain in quite a long period. This is like an intelligent manager of a baseball team in the Major League in US. Not intelligent managers try to win every game they fight. In order to get the championship of the season, however, no need to win all the games. Sometimes it is necessary to lose a game to win the next two, for example.

How We Incorporate These Additional Factors? Hence we need to add factors A reflecting these very human aspect.

$$V_{\mu}^{\pi} = A \sum_{j=1}^{n} \sum_{i=1}^{m} F(a_{ij}) \cdot G(a_{ij}) \cdot H(a_{ij}) \cdot L(a_{ij}) \cdot r_{ij}. \tag{12}$$

Kurzweil wrote, "You can have a human relationship with computers by 15 years from now." Though others are not so optimistic, like Larry Page, cofounder of Google, who wrote *"Despite a fantasy in the mid-20th century, nobody has yet figured out how to make a robot that can think*," [20].

Anyway, our concern is not specifically stock market forecasting but more general evaluation of machine intelligence for a various kind of specific tasks, to name a few, such as cyber security monitoring, forest fire monitoring, evaluation of essay or something else. How intelligent they will be indeed? With a human level or higher?

Maureen Dowd, journalist who won the Pulitzer, wrote "When I say about human levels, I'm talking about emotional intelligence, the ability to tell a joke, to be funny, to be romantic, to be loving, to be sexy, that is the cutting edge of human intelligence." [2]

She might say "I will buy Apple's stock because I love Mackintosh." Then could we have a measurement of such a level of machine intelligence, other than the Turing test [23]?

Acknowledgments The author greatly thanks Professor Jerzy Pejaś and the other organizing committee members of this conference for constantly inviting me to a series of this conference with wonderful atmosphere in a beautiful small town in Poland along the Baltic Sea. To take part in this conference has always been giving me a big motivation to proceed my reflection to what is machine intelligence.

References

1. Chaitin, G.J.: Gödel's theorem and information. Theor. Phys. **21**(12), 941–954 (1982)
2. Dowd, M.: Silicon valley Sharknado. Editorial in NYT on 08 July 2014
3. Ehrentreich, N.: A corrected version of the Santa Fe institute artificial stock market model (2004)
4. Floreano, D., et al.: Evolutionary robots with on-line self-organization and behavioral fitness. Neural Netw. **13**(4–5), 431–443 (2000)
5. Hasanhodzic, J. et al.: Is It Real, or Is It Randomized?: A Financial Turing Test, Cornell University Library (2010). arXiv:1002.4592
6. Hofstadter, D.: Gödel, Escher, Bach: An Eternal Golden Braid. Basic Books (1979)
7. Hofstadter, D.: http://afflictor.com/tag/douglas-hofstadter/ (2014)
8. Imada, A.: Intelligent machine? Then how intelligent? In: International Multi-Conference: Advanced Computer Systems (2012)
9. Ke, J., et al.: Modeling and simulation of the artificial stock market trading system. Appl. Math. Inf. Sci. **7**(4), 1599–1607 (2013)
10. Kim, H.-J., et al.: Weighted scale-free network in financial correlations. J. Phys. Soc. Jpn. **71**(9), 2131–2136 (2002)
11. Kluger, J.: Inside The Minds of Animals–TIME. Also available at http://content.time.com/time/magazine/article/0,9171/2008867,00.html (2010)
12. Kobos, M. et al.: Artificial intelligence methods in stock index prediction with the use of newspaper articles. In: Foundations of Control and Management Sciences, vol. 9, pp. 67–77. Publishing House of Poznan, University of Technology (2008)
13. Kurzweil, R.: Exponential Finance Conference. New York (2014)
14. Kurzweil, R.: The Observer of London. Also available at http://news.genius.com/Ray-kurzweil-on-google-and-the-singularity-annotated#note-2835513 (2014)
15. Kwon, Y-K.: Stock prediction based on financial correlation. In: Proceedings of the Annual Conference on Genetic and Evolutionary Computation, pp. 2061–2066 (2005)

16. Laloux, L., et al.: Noise dressing of financial correlation matrices. Phys. Rev. Lett. **83**, 1467–1469 (1999)
17. LeBaron, B.: Building the Santa Fe artificial stock market (2002)
18. Legg, S., Hutter, M.: Universal intelligence: a definition of machine intelligence. Minds. Mach. **17**(4), 391–444 (2007)
19. Malkiel, B.G.: A Random Walk Down Wall Street: The Time-Tested Strategy for Successful Investing. W. W. Norton & Company (2007)
20. Page, L.: Silicon Valley Summit (2014)
21. Plerou, V., et al.: Universal and nonuniversal properties of cross correlations in financial time series. Phys. Rev. Lett. **83**(7), 1471–1474 (1999)
22. Smith, W. D.: Mathematical definition of intelligence (and consequences). Available at http://math.temple.edu/wds/homepage/works.html (2006)
23. Turing, A.M.: Computing machinery and intelligence. Mind **59**(236), 433–460 (1948). Also available at http://www.loebner.net/Prizef/TuringArticle.html
24. Yixian, F. et al.: The stock index forecast based on dynamic recurrent neural network trained with GA. In: Proceedings of the 20th Pacific Asia Conference on Language, Information and Computation, pp. 319–323 (2006)
25. Zhang, J.: Modeling the influence of social networks in an artificial stock market. In: International Workshop on Agent-Mediated Electronic Commerce and Trading Agents Design and Analysis (2014)

Boosted Classifiers for Antitank Mine Detection in C-Scans from Ground-Penetrating Radar

Przemysław Klęsk, Mariusz Kapruziak and Bogdan Olech

Abstract We investigate the problem of automatic antitank mine detection. Subject to detection are 3D images, so-called C-scans, generated by a GPR (Ground-Penetrating Radar) system of our construction. In the paper we focus on *boosting* as a machine learning approach well suited for large-scale data such as GPR data. We compare five variants of weak learners with real-valued responses trained by the same boosting scheme. Three of the variants are single-feature-based learners that differ in the way they approximate class conditional distributions. The two remaining variants are shallow decision trees, respectively, with four and eight terminal nodes, introducing joint-feature conditionals.

Keywords Boosting · Weak learners · Ground Penetrating Radar · C-scans · Landmine detection

This project has been partially financed by the Polish Ministry of Science and Higher Education; agreement no. 0091/R/TOO/2010/12 for R&D project no. 0 R00 0091 12, dated on 30.11.2010, carried out by a consortium of Military Institute of Armament Technology in Zielonka, and Autocomp Management in Szczecin, Poland.

P. Klęsk (✉) · M. Kapruziak · B. Olech
Faculty of Computer Science, West Pomeranian University of Technology,
ul. Żołnierska 49, 71-210 Szczecin, Poland
e-mail: pklesk@wi.zut.edu.pl

M. Kapruziak
e-mail: mkapruziak@wi.zut.edu.pl

B. Olech
e-mail: bolech@wi.zut.edu.pl

M. Kapruziak · B. Olech
Autocomp Management, ul. Władysława IV 1, 70-651 Szczecin, Poland

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_2

1 Introduction

Postconflict areas are typically contaminated by explosive remnants such as unexploded ordnance (UXO) and landmines, both antipersonel (AP) and antitank (AT). United Nations report that about 80 countries are burdened with this problem (see http://www.un.org/en/globalissues/demining/). The process of bringing such areas back to normal use is slow and expensive [18].

As regards the demining technology, up to recently the electromagnetic induction (EMI) metal detector, often manual, has been the most frequent tool. Unfortunately, it is of little use for mines with low metal content (plastic, wooden, or glass mines), and moreover has a high false alarm rate (FAR)—it is estimated that about 10^3 false positives go on average with each correctly detected mine. The last decade brought a progress in the technology of Ground-Penetrating Radar, a method working with frequencies up to 20 GHz which potentially enables for high-resolution imaging and "seeing" nonmetalic objects. GPR-based mine detection is still far from ideal, as GPR is susceptible to all kinds of inhomogeneities and clutter in the ground (stones, roots, bricks, water pockets, etc.) and hence produces some false alarms as well, but results reported in the literature are much superior than the ones for EMI detectors [7, 18].

1.1 GPR Images and Mine Detection Approaches

A C-scan is a 3D image built up as a collection (taken over some area) of A-scans, i.e., one-dimensional signals generated by GPR over an axis directed into the ground (time axis t). A boundary of any buried object with dielectric properties different from the medium (soil) leaves a mark in an A-scan. As the GPR system moves across track (x-axis) or along track (y-axis), while taking successive A-scans, the marks generate a *hyperbola* trace in the image. Hence, an object seen in a C-scan resembles a combination of *hyperboloids*. The detection task is therefore about distinguishing mine-related hyperboloids from others. In the Fig. 1 we show detection examples in C-scans collected by our GPR vehicle.

Mine detection approaches met in the literature are often based on machine learning, and typically focus on two aspects: features extraction and learning algorithms. In majority of cases, extraction of features is done from preselected 2D projections (B-scans or time slices), and rather seldom directly from C-scans. Some authors try to reconstruct the physical features from an image (e.g., depth of burial, radius, height, dielectric permittivity of the ground) [19], however, the majority of works focuses on simpler and purely image-oriented features. By that meant are, e.g., histograms, statistical moments, principal components (PCA), structure-based features, edge, or shape-related descriptors [4, 9, 16].

In mentioned works, commonly a fairly small number of features is used to train classifiers—from several up to several tens. A notable exception can be met in [16]

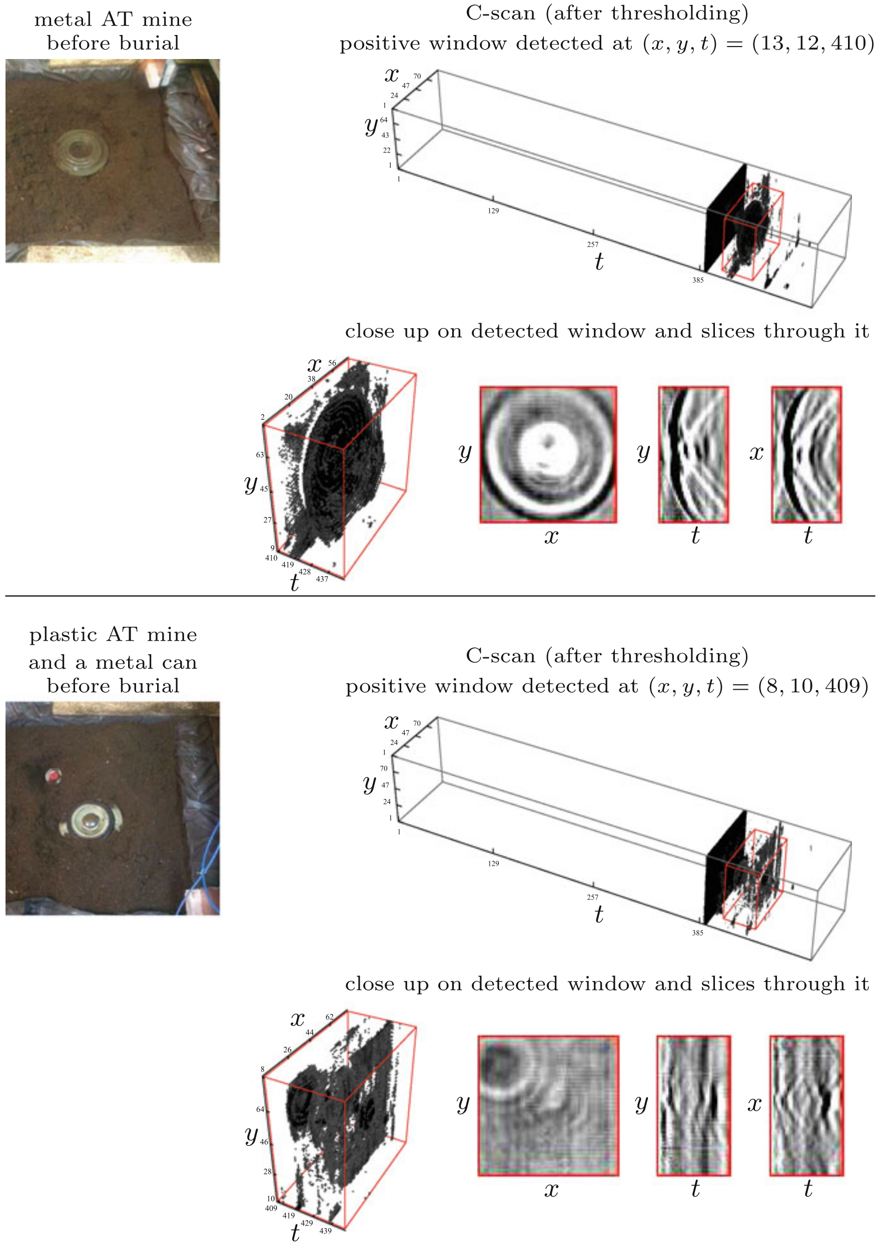


Fig. 1 Two examples of successful detections of metal AT mine (*upper* part) and plastic AT mine (*bottom* part). Input C-scans taken over approx. 1 m^2 area

due to Torrione and Collins, where authors adopted the *Texture Feature Coding Method* [6]. For given pixel of a C-scan they consider 13 difference—vectors crossing it within the 3×3 neighborhood. The vectors are then quantized into: {drop, no-change, growth}, and a codebook of textures is formed. By repeating such operations on many image fragments, potentially thousands of features can be generated. The authors report to limit themselves to 560 at maximum.

As regards the learning algorithms, quite many of them have been tried out in mine detection applications, e.g.: Naive Bayes and LVQ [2], neural networks [4], HMMs [4, 11], SVMs [5], RVMs [16]. It is difficult to compare fairly the results due to different experimental settings (types of mines used, soil types, amount of clutter, weather, etc.), nevertheless, to give a general view on recent achievements (e.g., [4, 5, 11, 14, 16])—commonly reported are detection rates (sensitivities) ranging approximately from 93 to 98 % with FARs typically not greater than $0.05\,\mathrm{FA}/\mathrm{m}^2$, and reported AUC[1] measures ranging within $95-99.5\,\%$.

1.2 Boosting and Our Motivation

The idea of boosting, proposed first by Schapire [13], is considered to be a crucial discovery of recent years in machine learning. Boosting is a meta-method. It works by sequentially applying some simple learning algorithm to reweighted versions of the training data, thus forming an ensemble of partial classifiers. The final response is taken as a majority or weighted vote. Known properties of boosting are its: (1) suitability for large-scale data, (2) ability to automatically select relevant features, and (3) robustness to overfitting (good generalization)—practical applications indicate that as new classifiers are added the test error decreases and then levels off, instead of ultimately increasing. Friedman, Hastie, and Tibshirani [3] have demonstrated that boosting can be viewed and understood as an *additive model* for *logistic regression*, explaining some of the “mystery" why boosting works well (see notes in the Appendix A).

In the context of mine detection, boosting might be of importance if one decides for a “mass attack" approach consisting in generating huge sets of simple features (e.g., of size $\sim 10^4$) from GPR images and letting the algorithm discover a smaller relevant subset (e.g., of size $\sim 10^2$). Note that such an approach worked very well in face detection applications, where ideas due to Viola and Jones [17] based on multiple Haar-like features are becoming a standard. Apart from our previous work [8] we have managed to come across only two papers [14, 15] on mine detection with the traditional AdaBoost being the learning algorithm.[2]

In this paper we apply the mentioned approach (at the training stage we generate over 15 thousand features—being 3D Fourier moments) and our main focus is to compare the performance of several variants of weak learners.

[1] Area under Receiver Operating Characteristic curve.

[2] Yet, the authors in [14, 15] consider quite small sets of about 20 features.

2 Boosting and Weak Learners to Be Compared

2.1 Notation

Let $\{(\mathbf{x}_i, y_i)\}_{i=1,\ldots,m}$ denote the set of training examples described by vectors of features $\mathbf{x}_i = (x_{i1}, \ldots, x_{id}) \in \mathbb{R}^d$ and class labels $y_i \in \{-1, 1\}$. In our context, $\mathbf{x}_i$ vectors represent features extracted from a 3D window being a part of a C-scan, and the positive class ($y_i = 1$) informs about a mine present in the window.

2.2 Boosting Scheme for Real-Valued Weak Learners

Beneath, we present a general boosting scheme in which weak learners are assumed to have real-valued responses (*real boost*). The number of rounds is denoted by K, weak classifiers by f_k, and the ensemble by F.

1. Start with uniform weights on data examples $w_i := 1/m$, $i = 1, \ldots, m$.
2. For $k = 1, \ldots, K$ repeat:

   2.1. Train a new weak classifier f_k using weights w_i on the training data with the goal to minimize the *exponential criterion* $\sum_{i=1}^{m} w_i e^{-y_i f_k(\mathbf{x}_i)}$, or equivalently so that $f_k(\mathbf{x})$ is an approximation of half the logit transform:

$$f_k(\mathbf{x}) := \frac{1}{2} \log \left(\widehat{P}_w(y = 1|\mathbf{x}) \Big/ \widehat{P}_w(y = -1|\mathbf{x}) \right). \tag{1}$$

   2.2. Update the weights:

$$Z_k := \sum\nolimits_{i=1}^{m} w_i e^{-y_i f_k(\mathbf{x}_i)}; \quad w_i := w_i e^{-y_i f_k(\mathbf{x}_i)}/Z_k, \; i = 1, \ldots, m. \tag{2}$$

3. The final (ensemble) classifier is $F(\mathbf{x}) := \sum_{k=1}^{K} f_k(\mathbf{x})$ with the decision returned according to sgn $F(\mathbf{x})$.

2.3 Compared Variants of Weak Learners

2.3.1 NormalRealBoost (NRB)

This learner iterates overall features x_j and approximates the conditional densities $p(x_j|y = \pm 1)$ by *normal* densities $\widehat{p}_w(x_j|y = \pm 1) = 1/\sqrt{2\pi\sigma_{j\pm}^2} e^{-(x_j-\mu_{j\pm})^2/(2\sigma_{j\pm}^2)}$ with means and variances calculated as:

$$\mu_{j-} = \sum_{\substack{i=1\\y_i=-1}}^{m} w_i x_{ij} \Big/ \sum_{\substack{i=1\\y_i=-1}}^{m} w_i, \qquad \mu_{j+} = \sum_{\substack{i=1\\y_i=1}}^{m} w_i x_{ij} \Big/ \sum_{\substack{i=1\\y_i=1}}^{m} w_i, \tag{3}$$

$$\sigma_{j-}^2 = \sum_{\substack{i=1\\y_i=-1}}^{m} w_i x_{ij}^2 \Big/ \sum_{\substack{i=1\\y_i=-1}}^{m} w_i - \mu_{j-}^2, \qquad \sigma_{j+}^2 = \sum_{\substack{i=1\\y_i=1}}^{m} w_i x_{ij}^2 \Big/ \sum_{\substack{i=1\\y_i=1}}^{m} w_i - \mu_{j+}^2. \tag{4}$$

By means of Bayes' theorem the response of this weak learner is calculated as:

$$f_k(\mathbf{x}) = \frac{1}{2} \log \frac{\widehat{p}_w(x_{j^*}|y=1)\widehat{P}_w(y=1)}{\widehat{p}_w(x_{j^*}|y=-1)\widehat{P}_w(y=-1)} \tag{5}$$

$$= \frac{1}{2}\left(\frac{(x_{j^*}-\mu_{j^*-})^2}{2\sigma_{j^*-}^2} - \frac{(x_{j^*}-\mu_{j^*+})^2}{2\sigma_{j^*+}^2} + \log\frac{\sigma_{j^*-}}{\sigma_{j^*+}} + \log\frac{\widehat{P}_w(y=1)}{\widehat{P}_w(y=-1)}\right), \tag{6}$$

where $\widehat{P}_w(y=\pm 1) = \sum_{\{i\,:\,y_i=\pm 1\}} w_i$ are current class probability estimates and j^* indicates the feature for which the exponential criterion is the smallest.

2.3.2 BinningRealBoost (BRB)

The idea behind this learner is similar to NRB, with the difference that conditional densities for each feature are approximated by *piecewise constant* functions instead of normals. Such functions are typically implemented by a binning mechanism, see e.g., [8, 12].

Suppose that $[a_1, a_2]$ represents the range of some features. Let B denote the wanted number of bins. For convenience we use bins of equal widths. The bin index $\beta(x) \in \{1, \ldots, B\}$ that an argument x belongs to is: $\beta(x) = \lceil B(x-a_1)/(a_2-a_1)\rceil$ for $a_1 < x \le a_2$; with border cases: $\beta(x)=1$ for $x \le a_1$ and $\beta(x)=B$ for $a_2 < x$. Let $\widehat{P}_w(y=-1, j \text{ in } b) = \sum_{\{i\,:\,y_i=-1,\ \beta(x_{ij})=b\}} w_i$ denote the estimated probability that an example is negative and its jth feature belongs to the bin b. Then, the response of this weak classifier using the best j^*th feature is:

$$f_k(\mathbf{x}) = \frac{1}{2} \log\left(\widehat{P}_w\left(y=1, j^* \text{ in } \beta(x_{j^*})\right) \Big/ \widehat{P}_w\left(-, j^* \text{ in } \beta(x_{j^*})\right)\right). \tag{7}$$

In our experiments we shall test two variants named BRB4 and BRB8 where the number of bins B is four and eight, respectively.

2.3.3 DecisionTreesRealBoost (DTRB)

This learner is based on the well-known CART algorithm [1]. Practical and theoretical evidences show that an ensemble of many shallow decision trees generalizes better than a deep single tree [3, 10].

In experiments we test two variants named DTRB4 and DTRB8 representing trees with four and eight terminal nodes, respectively. The generation of each tree has been carried out in a traditional fashion using Gini index as the impurity criterion. Not to repeat the whole algorithm, hereby we describe only the splits evaluation. Suppose we are at a certain stage of the recursion and the set of indices $\{i\}$ is restricted only to data examples falling into the given tree node. Quantities of interest for a candidate split on the jth feature and a value v are:

$$\begin{aligned} W(L) &= \sum_{\{i\,:\,x_{ij}<v\}} w_i, & W(R) &= \sum_{\{i\,:\,x_{ij}\geq v\}} w_i, \\ W(y{=}-1, L) &= \sum_{\{i\,:\,x_{ij}<v,\ y_i=-1\}} w_i, & W(y{=}1, L) &= \sum_{\{i\,:\,x_{ij}<v,\ y_i=1\}} w_i, \\ W(y{=}-1, R) &= \sum_{\{i\,:\,x_{ij}\geq v,\ y_i=-1\}} w_i, & W(y{=}1, R) &= \sum_{\{i\,:\,x_{ij}\geq v,\ y_i=1\}} w_i, \end{aligned} \tag{8}$$

where L and R denote, respectively, the left and the right part after the split is made, and the suitable probability estimates can be calculated as follows:

$$\begin{aligned} \widehat{P}_w(L) &= W(L)/\left(W(L)+W(R)\right), & \widehat{P}_w(R) &= W(R)/\left(W(L)+W(R)\right), \\ \widehat{P}_w(y{=}-1|L) &= W(y{=}-1, L)/W(L), & \widehat{P}_w(y{=}1|L) &= W(y{=}1, L)/W(L), \\ \widehat{P}_w(y{=}-1|R) &= W(y{=}-1, R)/W(R), & \widehat{P}_w(y{=}1|R) &= W(y{=}1, R)/W(R). \end{aligned} \tag{9}$$

Then, the expected Gini impurity of the split is:

$$\widehat{P}_{w(L)}\left(1-\widehat{P}_w^2(y{=}-1|L)-\widehat{P}_w^2(y{=}1|L)\right)+\widehat{P}_{w(R)}\left(1-\widehat{P}_w^2(y{=}-1|R)-\widehat{P}_w^2(y{=}1|R)\right). \tag{10}$$

Each terminal node of a tree responds with a real value in accordance with half the logit function (1), i.e., $\frac{1}{2}\log(\sum_{\{i\,:\,y_i=1\}} w_i / \sum_{\{i\,:\,y_i=-1\}} w_i)$.

3 Experiments

3.1 Learning Material

Our GPR-equipped vehicle (see the Fig. 2) was initially tested in outdoor conditions (over peat, garden, gravel, sand soils) and proved to produce sufficiently clear images. For convenience, the main learning material was collected in indoor conditions on a laboratory test stand with garden soil. Scanned scenes involved two types of AT mines (metal and plastic) and a variety of disruptive objects (cans, boxes, bricks, discs, shafts, cables). We experimented with different burial depths, lean angles, surface variations, "mine imitations" set up by suitable arrangements of nonmine objects; some scene examples are depicted in the Fig. 2c.

The final material consisted of 210 high-resolution C-scans (1 cm both across and along track, 512 time samples), each taken from an area of $\approx$1 m^2. The material included: 70 scans with the metal AT mine (and possibly other objects), 70 scans with the plastic AT mine (and possibly other objects), 70 scans with nonmine objects only. The scans were then divided into train (80 %) and test (20 %) parts, with the test part consisting of: 14 metal mine scans, 14 plastic mine scans, and 14 scans with nonmines. Finally, all scans were traversed by a sampling 3D window of widths $(w_x, w_y, w_t) = (63, 63, 35)$ and $(w_x, w_y, w_t) = (72, 72, 40)$—two passes over each C-scan, and a data set of positive and negative window examples was created for supervised learning. We have memorized all positive windows, whereas negative windows were undersampled. This resulted in a $\approx$ 4 GB train set with about 2.1×10^3 positives and 6.8×10^4 negatives; and a $\approx$ 1 GB test set with 5.5×10^2 positives and

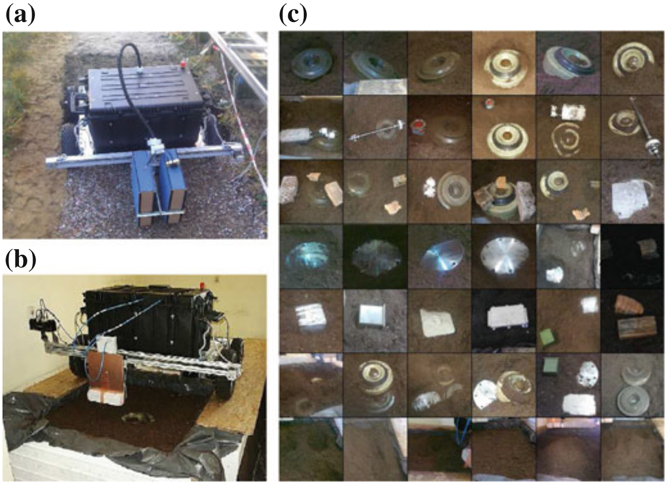


Fig. 2 GPR vehicle of our construction shown outdoor (**a**) and on a laboratory test stand (**b**). Examples of scanned scenes and variations (**c**)

1.7×10^4 negatives. Each window example was described by 15,625 shape-related features.

3.2 Features Extraction

To extract features we applied *piecewise 3D Fourier approximations* (of low orders) with respect to image windows. Each window was divided into a regular grid of 125 cuboids ($5 \times 5 \times 5$), and for each cuboid, spanning in general from (x_1, y_1, t_1) to (x_2, y_2, t_2), Fourier coefficients (3D moments) were calculated as:

$$c_{k_x,k_y,k_t} = \frac{\sum\limits_{x=x_1}^{x_2}\sum\limits_{y=y_1}^{y_2}\sum\limits_{t=t_1}^{t_2} i(x,y,t)e^{-2\pi i\left(k_x\frac{x-x_1}{x_2-x_1+1}+k_y\frac{y-y_1}{y_2-y_1+1}+k_t\frac{t-t_1}{t_2-t_1+1}\right)}}{(x_2-x_1+1)(y_2-y_1+1)(t_2-t_1+1)}, \qquad (11)$$

where: $-n \leq k_x, k_y, k_t \leq n$ indicate harmonic orders (variable-wise) of the coefficient with n being the maximum, $i(x, y, t)$ is the pixel intensity at a particular point of the C-scan, and $i = \sqrt{-1}$ is the imaginary unit. Then, real and imaginary parts of the coefficients were taken as the features.[3]

Due to the well-known symmetry property: $\mathrm{Re}\, c_{k_x,k_y,k_t} = \mathrm{Re}\, c_{-k_x,-k_y,-k_t}$, $\mathrm{Im}\, c_{k_x,k_y,k_t} = -\mathrm{Im}\, c_{-k_x,-k_y,-k_t}$, and the fact that $\mathrm{Im}\, c_{0,0,0} = 0$, it is easy to check that the number of distinct real and imaginary parts is $(2n+1)^3$. For computational reasons and after some initial experimentation, we imposed a low-order $n = 2$ of approximation, and therefore the effective number of features, taking into account all cuboids, was $d = 125 \cdot (2n + 1^3) = 15{,}625$.

3.3 Results of Experiments

All results are reported with a distinction between detectors trained to detect only metal or only plastic AT mines ("Metal"/"Plastic detectors" for short).

Figures 3, 4 and the Table 1 summarize results obtained at the "window-level of detail" for the 1 GB test set (5.5×10^2 positive windows, 1.7×10^4 negative windows). It means that each window sampled from test C-scans is treated as separate object under detection. We remark that a positive target (mine) in the image is represented not by a single window, but a concentrated cluster of multiple windows[4] differing by small pixel shifts due to a certain tolerance. In the Table 1, the AUC_α notation stands

[3] At the moment of writing this manuscript, we submitted a paper to the IEEE Trans. on Geoscience and Remote Sensing journal, proposing a technique for fast calculation of moments (11) for successive windows via multiple integral images.

[4] Even about 100 windows for a dense image traversal, e.g., with 1 pixel shifts.

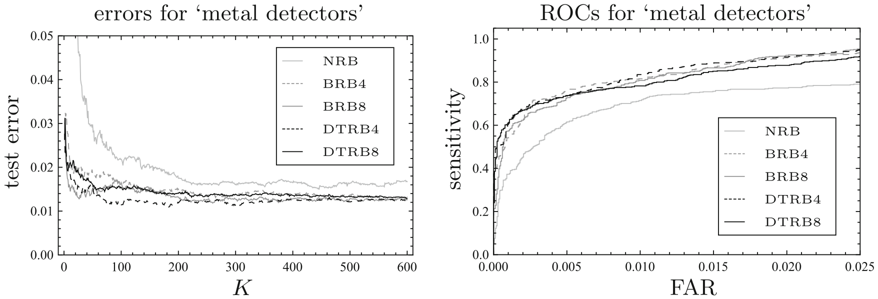


Fig. 3 Test errors and ROC curves for "metal detectors"

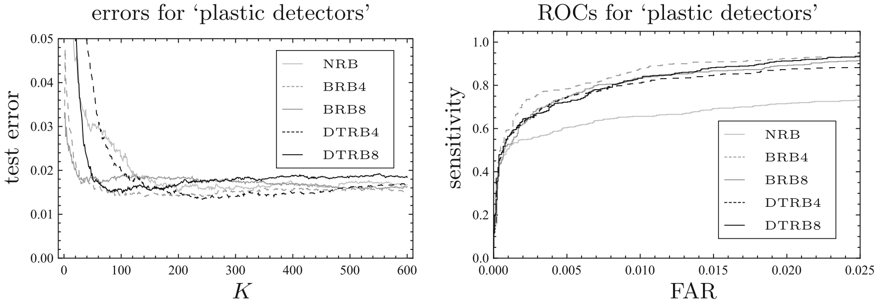


Fig. 4 Test errors and ROC curves for "plastic detectors"

for the normalized area under ROC obtained up to the FAR equal α. This shows how fast the ROC grows in its initial stage.

Results at the "image-level of detail" (for whole C-scans) are reported in the Table 2. For these results, decisions of detectors where thresholded, i.e., $\text{sgn}(F(\mathbf{x}) - \theta)$, with thresholds θ chosen from ROCs to correspond to the 0.5 sensitivity. This operation was done in order to significantly decrease the number of false alarms in images, but still keeping high detectivity. Note that if a target is represented, e.g., by a cluster of 10 windows, then the probability that at least one of them will be "turned on" (given the sensitivity 0.5 per each window) is at least $1 - 0.5^{10} \geq 0.999$, assuming an independence model for simplification.

The following summarizing comments on the results can be formulated. Obviously, our data set is strongly imbalanced, favoring the negative class, and so are classification results. If a zero-rule classifier is considered as a reference (accuracies 97, 96.80 % respectively for metal and plastic AT mines) then all detectors surpassed it. The NRB classifier performed clearly worst, both as the "metal" and the "plastic detector." It is particularly apparent in the ROC curves. Other classifiers exhibited only minor differences in ROCs among each other. Looking at results for the whole C-scans (see Table 2), the NRB detectors produced an unacceptably high number of false alarms (17/42 and 20/42). Both types of decision trees—DTRB4, DTRB8—turned out to be the best as "Metal detectors," dominating the

Table 1 Test results for detectors at “windows level of detail"

Name	Acc. (%)	Sens. (%)	FAR	AUC (%)	$AUC_{0.01}$(%)	$AUC_{0.001}$(%)	FAR at Sens. 1/2	FAR at Sens. 1/3
“Metal detectors”								
Zero-rule	97.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NRB	98.32	56.50	3.78×10^{-3}	87.62	56.61	26.49	2.70×10^{-3}	6.61×10^{-4}
BRB4	**98.76**	**65.44**	2.10×10^{-3}	99.43	71.92	47.39	7.81×10^{-4}	**0.00**
BRB8	98.68	61.75	1.74×10^{-3}	99.46	69.20	42.72	6.61×10^{-4}	3.00×10^{-4}
DTRB4	98.74	61.36	1.02×10^{-3}	**99.49**	**72.35**	52.11	3.60×10^{-4}	**0.00**
DTRB8	98.72	59.81	$\mathbf{7.81 \times 10^{-4}}$	99.43	71.33	**53.98**	$\mathbf{2.40 \times 10^{-4}}$	6.01×10^{-5}
“Plastic detectors”								
Zero-rule	96.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00
NRB	98.33	50.73	1.02×10^{-3}	82.81	58.18	38.37	9.00×10^{-4}	3.00×10^{-4}
BRB4	**98.47**	**54.36**	7.20×10^{-4}	99.30	**75.41**	**45.20**	$\mathbf{4.80 \times 10^{-4}}$	2.40×10^{-4}
BRB8	98.37	52.73	1.20×10^{-3}	99.01	70.75	42.34	7.20×10^{-4}	$\mathbf{1.80 \times 10^{-4}}$
DTRB4	98.31	49.09	6.60×10^{-4}	98.25	70.11	42.93	6.60×10^{-4}	$\mathbf{1.80 \times 10^{-4}}$
DTRB8	98.16	43.45	$\mathbf{3.60 \times 10^{-4}}$	**99.40**	70.00	43.12	6.60×10^{-4}	2.40×10^{-4}

Table 2 Test results for detectors at “images level of detail"

Object type	NRB		BRB4		BRB8		DTRB4		DTRB8	
	Detected as metal mine	Side false alarms	Detected as metal mine	Side false alarms	Detected as metal mine	Side false alarms	Detected as metal mine	Side false alarms	Detected as metal mine	Side false alarms
“Metal detectors”										
Metal mine	11/14	7/14	14/14	1/14	13/14	0/14	14/14	1/14	13/14	0/14
Plastic mine	1/14	3/14	0/14	2/14	0/14	1/14	0/14	0/14	0/14	0/14
Other object	4/14	2/14	4/14	1/14	3/14	0/14	2/14	1/14	1/14	0/14
Sens.	11/14 = 78.57%		14/14 = 100.86%		13/14 = 92.86%		**14/14 = 100.0%**		**13/14 = 92.86%**	
FAR	17/42 = 40.48%		8/42 = 19.05%		4/42 = 9.52%		**3/42 = 7.14%**		**2/42 = 4.76%**	
“Plastic detectors”										
Metal mine	3/14	6/14	1/14	3/14	0/14	0/14	1/14	3/14	0/0	4/14
Plastic mine	13/14	0/14	12/14	4/14	13/14	1/14	13/14	0/14	13/14	1/14
Other object	7/14	4/14	1/14	2/14	3/14	1/14	4/14	2/14	3/14	1/14
Sens.	13/14 = 92.86%		12/14 = 85.71%		**13/14 = 92.86%**		13/14 = 92.86%		0/0 = 0.0%	
FAR	20/42 = 47.62%		11/42 = 26.19%		**5/42 = 11.90%**		10/42 = 23.81%		9/42 = 21.42%	

results of other classifiers. In particular, the DTRB4 achieved $14/14 = 100\,\%$ sensitivity with $3/42 = 7.14\,\%$ FAR. This can be also explained by looking at ROC/AUC characteristics—both types of trees exhibited highest $\text{AUC}_{0.001}$. As regards “Plastic detectors,” the results were in general slightly worse. The eight bins real boost—BRB8—surpassed other classifiers, obtaining $13/14 = 92.86\,\%$ sensitivity with $5/42 = 11.90\,\%$ FAR. Also, we think it is fair to remark that our material was rather difficult, containing many disruptive objects and scenes with mine resemblance. In a more natural test lane with many “empty" square meters, more appealing FAR rates could be achieved.

4 Conclusions

High-trust GPR systems for automatic mine detection require very large data sets to be collected and learned from. In our case study, we worked with a fairly limited collection of 210 C-scans (from $\approx$1 m^2 each), but we generated a large data set from it—with approximately 7×10^4 examples of 3D windows each described by over 15 thousand features.

Our experiments showed that a commonly applied single-feature weak learner which approximates class conditional probabilities by normal distributions (popularly known as “real boost," NRB in our notation) was significantly outperformed by learners only slightly more refined—based on binning and shallow decision trees. In particular, the NRB yielded the AUC measures of 87.6 and 82.8 % respectively for metal and plastic AT mines, whereas bins and trees (apart from one exception per eight cases) obtained AUCs greater than 99 %.

Appendix A: Notes on Boosting’s Connection to Logistic Regression

Following [3], we point out some important properties of boosting in the context of its connection to logistic regression and additive modeling.

Think of the unknown joint probability distribution (population) from which the data is drawn. Let $p(\mathbf{x}, y)$ denote its density function, which can also be expressed as $p(\mathbf{x}, y) = P(y|\mathbf{x})p(\mathbf{x})$. Now, consider the exponential criterion for some classifier function F defined as an expectation taken with respect to p:

$$\begin{aligned} Q_p(F) = \mathbb{E}_p\big(e^{-yF(\mathbf{x})}\big) &= \int_{\mathbf{x}} \sum_{y\in\{-1,1\}} e^{-yF(\mathbf{x})} p(\mathbf{x}, y)\,\mathbf{dx} \\ &= \int_{\mathbf{x}} \Big(P(y{=}-1|\mathbf{x})e^{F(\mathbf{x})} + P(y{=}1|\mathbf{x})e^{-F(\mathbf{x})}\Big) p(\mathbf{x})\,\mathbf{dx}. \end{aligned} \tag{12}$$

To minimize Q_p we demand that $\partial Q_p(F)/\partial F = 0$ (in fact it suffices to minimize the inner conditional expectation) and obtain the formula for the minimizer $F^*(\mathbf{x}) = 1/2 \log\left(P(y{=}1|\mathbf{x})/P(y{=}{-1}|\mathbf{x})\right)$ being half the logit transform, typical for logistic regression. If an algorithm could somehow immediately find the perfect function F^*, then the boosting procedure could be stopped after just one round. In practice, weak learners are rough approximations of F^* and multiple rounds are needed. Solving F^* for probability leads to a form of sigmoid:

$$P(y = 1|\mathbf{x}) = e^{2F^*(\mathbf{x})} \Big/ \left(1 + e^{2F^*(\mathbf{x})}\right) = 1 \Big/ \left(1 + e^{-2F^*(\mathbf{x})}\right), \tag{13}$$

and again the similarity to logistic regression can be seen. The two minor differences are: the factor of 2 in the exponent, and the fact that in the traditional logistic regression one approximates F^* by a linear model $F^*(\mathbf{x}) \approx a_0{+}a_1x_1{+}\cdots{+}a_nx_n$, whereas in boosting it is a linear combination of weak learners, i.e., $F^*(\mathbf{x}) \approx \sum_k f_k(\mathbf{x})$, so in general arbitrary but simple functions, each being possibly a function of all variables.

Think of the *error residuals* technique, known from regression modeling in general. In an additive model built sequentially, each successive piece of approximation “explains" some part of the target quantity and that part is subtracted from the target, so that future pieces can focus on residuals. Boosting’s reweighing scheme proceeds in an akin manner. Suppose we have fixed a partial model F and would like to update it, i.e., to have $F := F + f$. Recall the data-based reweighing formulas (2). Let us define their population-based counterparts:

$$Z = \int_{\mathbf{x}} \sum_{y\in\{-1,1\}} e^{-yF(\mathbf{x})} p(\mathbf{x}, y)\,\mathbf{dx}, \qquad w(\mathbf{x}, y) = e^{-yF(\mathbf{x})} p(\mathbf{x}, y)/Z. \tag{14}$$

Note that Z works as a normalizer but simultaneously is our exponential criterion value for the model so far—$Q_p(F)$. Now, consider the criterion at $F + f$:

$$\begin{aligned} Q_p(F{+}f) &= \int_{\mathbf{x}} \sum_{y\in\{-1,1\}} e^{-y(F(\mathbf{x})+f(\mathbf{x}))} p(\mathbf{x}, y)\,\mathbf{dx} \\ &= \int_{\mathbf{x}} \sum_{y\in\{-1,1\}} e^{-yf(\mathbf{x})}\, \underbrace{e^{-yF(\mathbf{x})} p(\mathbf{x}, y)/Z}_{w(\mathbf{x},y)}\, \mathbf{dx} \cdot Z{=}Q_w(f) \cdot Q_p(F). \end{aligned} \tag{15}$$

Therefore, to minimize $Q(F + f)$ it suffices to greedily minimize $Q_w(f)$. The distribution w indicates which “parts” of the target quantity are already explained.

References

1. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth & Brooks, Monterey (1984)
2. Cremer, F., et al.: Feature level fusion of polarimetric infrared and GPR data for landmine detection. In: Proceedings of EUDEM2-SCOT 2003, International Conference on Requirements and Technologies for the Detection, Removal and Neutralization of Landmines and UXO, vol. 2, pp. 638–642 (2003)
3. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Ann. Stat. **28**(2), 337–407 (2000)
4. Frigui, H., et al.: Context-dependent multisensor fusion and its application to land mine detection. IEEE Trans. Geosci. Remote Sens. **48**(6), 2528–2543 (2010)
5. Hamdi, A., Missaoui, O., Frigui, H.: An SVM classifier with HMM-based kernel for landmine detection using ground penetrating radar. In: IEEE International Geoscience and Remote Sensing Symposium (IGARSS), pp. 4196–4199 (2010)
6. Horng, M.H.: Texture feature coding method for texture classification. Opt. Eng. **42**(1), 228–238 (2002)
7. Jol, H.M.: Ground Penetrating Radar: Theory and Applications. Elsevier, Oxford (2009)
8. Klęsk, P., Godziuk, A., Kapruziak, M., Olech, B.: Landmine detection in 3D images from ground penetrating radar using Haar-like features. In: ICAISC 2013, Zakopane. Lecture Notes in Artificial Intelligence, vol. 7894, pp. 559–567. Springer, Berlin (2013)
9. Ligthart, E.E., Yarovoy, A.G., Roth, F., Ligthart, L.P.: Landmine detection in high resolution 3D GPR images. In: MIKON'04, pp. 423–426 (2004)
10. Mease, D., et al.: Boosted classification trees and class probability/quantile estimation. J. Mach. Learn. Res. **8**, 409–439 (2007)
11. Missaoui, O., Frigui, H., Gader, P.: Land-mine detection with ground-penetrating radar using multistream discrete hidden Markov models. IEEE Trans. Geosci. Remote Sens. **49**(6), 2080–2099 (2011)
12. Rasolzadeh, B., et al.: Response binning: improved weak classifiers for boosting. In: IEEE Intelligent Vehicles Symposium, pp. 344–349 (2006)
13. Schapire, R.E.: The strength of weak learnability. Mach. Learn. **5**, 197–227 (1990)
14. Shi, Y., et al.: Landmine detection using boosting classifiers with adaptive feature selection. In: IEEE 6th International Workshop on Advanced Ground Penetrating Radar (IWAGPR), pp. 1–5 (2011)
15. Sun, Y., Li, J.: Adaptive learning approach to landmine detection. IEEE Trans. Aerosp. Electron. Syst. **41**(3), 973–985 (2005)
16. Torrione, P., Collins, L.M.: Texture features for Antitank landmine detection using ground penetrating radar. IEEE Trans. Geosci. Remote Sens. **45**(7), 2374–2382 (2007)
17. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2001), pp. 511–518 (2001)
18. Yarovoy, A.: Landmine and unexploded ordnance detection and classification with ground penetrating radar. In: Jol, H.M. (ed.) Ground Penetrating Radar: Theory and Applications, pp. 445–478. Elsevier, Oxford (2009)
19. Yarovoy, A.G., Kovalenko, V., Fogar, L.P.: Impact of ground clutter on buried object detection by ground penetrating radar. In: International Geoscience and Remote Sensing Symposium, pp. 755–777 (2003)

Detection Accuracy of the Temporary State of Complex Signals Using Phase-Frequency Tracking Methods with Equilibrium and Non-equilibrium Processing

Alexander Kochegurov, Elena Kochegurova and Natalya Kupina

Abstract This report proposes the phase-frequency tracking methods of complex signals based on optimal and suboptimal processing of PFC and group delay functions (FGZ). At the same time, the cases of correlated and uncorrelated samples of PFC and FGZ are analyzed herein. It is demonstrated that the correlation in the samples does not change the processing structure, but only the weight factors are changed. The phase-frequency algorithms with equilibrium and nonequilibrium processing are developed based on the proposed methods. It is shown that the transition to equilibrium processing enables a significant level down the requirements to a priori information about the properties of the useful signal, while nonequilibrium processing increases the resolution of the signals significantly. The conducted analytical argument and results of the simulation experiments on the produced simulated wave fields have testified that at propagation of the complex signals in dispersive media, these algorithms can assure rather high detection accuracy of the signal temporary state, even when the signal-to-noise ratio is close to 1. The results of the simulation experiments are justified by the real data obtained in processing the seismic wave fields.

Keywords Phase-frequency characteristic · Equilibrium and Non-equilibrium processing · Time position signal

1 Introduction

In the performance of a wide range of tasks in such areas as radio detection, navigation, and communication and geophysics, there arise problems related to improper detection accuracy of the temporary state of complex signals, which is primarily conditioned by the presence of interference ranges of useful signals, dispersion nature of

A. Kochegurov · E. Kochegurov (✉) · N. Kupina
National Research Tomsk Polytechnic University, Tomsk, Russia
e-mail: kocheg@mail.ru

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_3

their propagation medium, and the presence of intense incoherent noise [1]. Under such conditions, the customary correlation methods for detecting the temporary state of the complex signals prove to be very inefficient [2, 3] and require using the innovative approaches that make use of less a priori information about the properties of the useful signal while working sufficiently stable at signal propagation in the dispersive and absorption media. In this respect, the most promising are the phase-frequency methods of complex signals tracking, based on the effective processing of the information extracted from the phase-frequency characteristic (PFC) of the signal [4]. The prerequisite for their successful implementation is the fact that the phase of the signal, more precisely the complex law of variation of its PFC, provides information that allows more efficient detection of the signal and measurement of its temporary state under high-amplitude noise [5].

2 Methods Development

The most common solution for determining the signal's temporary state is to evaluate one of the nonenergy parameters of the normal random process and not account for the specificity of the time parameter. At the same time, the view of the signal's temporary state in exponential basis is totally determined by its phase-frequency characteristic (PFC). Therefore, the optimal process of the PFC signal implements the optimal method for determining its temporary state [6].

Let the given mixture of additive deterministic signal $s(t)$ and Gaussian noise $n(t)$ be:

$$x(t, \tau) = s(t - \tau) + n(t), \tag{1}$$

where τ is the signal's temporary state.

It is required to analyze the phase-frequency algorithms to investigate the accuracy of the complex signals' temporary provisions with the use of equilibrium and nonequilibrium processes.

As it is known, the optimal method for determining the phase of signal's temporary state observed on the background of Gaussian noise is implemented as searching for maximum likelihood of the function [5]:

$$L(t) = \sum_{k=1}^{m} \delta\,(\omega_k) \cdot \cos\,(\Delta\phi(\omega_k) - \omega_k t). \tag{2}$$

where $\Delta\phi(\omega_k) = \phi_x\,(\omega_k) - \phi_s\,(\omega_k)$—the deviation of the signal's phase from the phase spectrum of the mixture of signal and noise;

$\delta(\omega_k) = \frac{A(\omega_k)}{\sigma(\omega_k)}$—the peak signal to noise ratio at the frequency ω_k, and m—the number of analyzed frequency components.

It is not difficult to show that in case of a strong signal from the expression (1), the temporary state of the signal [7] can be directly accessed as

$$\tau_{\text{opt}} = \frac{\sum_{k=1}^{m} \delta^2\,(\omega_k) \cdot \omega_k \cdot \Delta\phi(\omega_k)}{\sum_{k=1}^{m} \delta^2\,(\omega_k) \cdot \omega_k^2} \tag{3}$$

In this case, the variance of (3) is:

$$D\left(\tau_{opt}\right) = \left[\sum_{k=1}^{m} \delta^2\,(\omega_k) \cdot \omega_k^2\right]^{-1}. \tag{4}$$

If the values in the sample phase response are dependent among themselves, the optimal estimate of the temporary state of a strong signal in the integral form is as follows [7]:

$$\tau_{\text{opt}} = \frac{\int_{\Omega} V(\omega) \cdot [\phi_x(\omega_k) - \phi_s(\omega_k)]\, d\,\omega}{\int_{\Omega} V(\omega) d\omega}. \tag{5}$$

where

$$V(\omega) = \int_{\Omega} R^{-1}(\omega, \omega') \cdot \omega' d\omega'. \tag{6}$$

$R(\omega, \omega')$—the positive definite matrix, consisting of the elements of the frequency correlation function PFC mixture; Ω—the analyzed frequency band, and the variance of (5) is

$$D\left(\tau_{\text{opt}}\right) = \left[\int_{\Omega}\int_{\Omega} R^{-1}(\omega, \omega') \cdot \omega \cdot \omega' d\omega\, d\omega'\right]^{-1}. \tag{7}$$

Comparison of (3) and (5) shows that when measuring the temporary state of a strong signal correlation values phase response in the sample mixture leads only to a change in the weighting procedures for handling phase response. Therefore, the expression (2) can be used to estimate the state of the signal, both uncorrelated and correlated sample PFC for any signal-to-noise ratio. Also, it can be shown in (3) that in the case of a weak signal and uncorrelated variance of the sample PFC, the time expression of the signal is:

$$D\left(\tau_{\text{opt}}\right) \approx \frac{4}{\pi} \cdot \left[\sum_{k=1}^{m} \delta^2\left(\omega_k\right) \cdot \omega_k^2\right]^{-1}. \tag{8}$$

In practice, the optimal estimates of the signal's temporary state by maximizing the expression (2) is often not possible, since the distribution of the signal/noise ratio in the test band Ω, forming weights in (2) is usually unknown. It is therefore proposed to use the so-called phase frequency algorithms with equilibrium and nonequilibrium processing.These algorithms can be obtained from the optimal method by replacing in (2) the weight function $\delta\left(\omega_k\right)$ with other specially selected functions. In general, the likelihood function (criterion time position signal) for these algorithms can be written as:

$$L(t) = \sum_{k=1}^{m} w\left(\omega_k\right) \cdot \cos\left(\phi\left(\omega_k, t\right)\right). \tag{9}$$

where $w(\omega_k)$ is the frequency weighting function, the form of which depends on the implemented phase-frequency algorithm.

For the equilibrium algorithm weighting function $w(\omega_k)$ has the value of unity in the whole frequency band. The algorithm of nonequilibrium processing $w(\omega_k)$ can have any form, but numerous experiments with different weight functions show [8] that the one with a triangular weight function is the best:

$$w\left(\omega\right) = \frac{4}{3\omega_m} \begin{cases} 0, & \omega < \omega_L \\ \frac{2}{\omega_m}\left(\omega - \omega_L\right), & \omega_L < \omega < \omega_m \\ -\frac{1}{\omega_m}\left(\omega - \omega_H\right), & \omega_m < \omega < \omega_H \end{cases}, \tag{10}$$

where ω_H and ω_L are, respectively, the higher and lower frequency bounds constraining $w\left(\omega_k\right)$, and $w\left(\omega_m\right)$ is the peak frequency of $w\left(\omega_k\right)$; $\omega_m = 2\omega_L$; $\omega_H = 2\omega_m$.

In the general case, Eq. (9) can be a picking result, because there is some analogy between estimating times in the conventional algorithms and the lowpass filtering. Indeed, the likelihood function Eq. (9) is an inverse discrete Fourier transform of the digitally filtered initial data, with the bandpass [7]:

$$H\left(\omega_k\right) = \frac{w\left(\omega_k\right)}{|X(k)|},\ k = \overline{1, m}, \tag{11}$$

where $|X(k)|$ is the amplitude response.

As follows from (11), this bandpass filtering first straightens the amplitude spectrum and then weighs it with the specified weight coefficients, while the phase patterns in the initial record remain the same. Straightening the amplitude response in the case of a linear phase response is known [9] to compress the signal and, hence, make it better resolved in the record. Furthermore, this kind of filter makes it possible to manage the bandpass frequency by specifying the weight coefficients $w\left(\omega_k\right)$ and thus either to strengthen or to weaken different frequency components of the signal.

It is not difficult to show that for a strong signal and uncorrelated variance of the sample PFC, the time position for the equilibrium processing algorithms is:

$$D\left(\tau\right)=\frac{\sum_{k=1}^{m}\frac{\omega_k^2}{\delta^2\left(\omega_k\right)}}{\left(\sum_{k=1}^{m}\omega_k^2\right)^2}\tag{12}$$

And for nonequilibrium processing algorithms,

$$D\left(\tau\right)=\frac{\sum_{k=1}^{m}\frac{w^4\left(\omega_k\right)\cdot\omega_k^2}{\delta^2\left(\omega_k\right)}}{\left(\sum_{k=1}^{m}w^2\left(\omega_k\right)\cdot\omega_k^2\right)^2}.\tag{13}$$

Comparison of (12) and (13) with (4) the transition to the equilibrium and nonequilibrium algorithms reduces the accuracy of the estimates, but this approach requires much less a priori information about the signal to be recorded, namely only the information about the values of its PFC in analyzed frequency band.

At signal propagation in dispersive medium, the temporal position of the signal is determined by its group delay [10]. In such event, the optimal method to determine the temporal position of the signals is realized in the form of the search procedure of the maximum of the plausibility function of the group delay [11]:

$$lnI\left(\tau\right)=\sum_{k=1}^{m}\gamma\left(\omega_k\right)\cos\left(\omega_k\left(\Delta t_{gr}\left(\omega_k\right)-\tau\right)\right)\tag{14}$$

where, $\Delta t_{gr}(\omega_k)=t_{gr}^{x}(\omega_k)-t_{gr}^{s}(\omega_k)$

$t_{gr}^{x}(\omega)$—function of the group delay (FGD) of the noisy signal mixture;

$t_{gr}^{s}(\omega)$—signal FGD;

$\gamma(\omega_k)$—signal/noise ratio in the derivatives at the frequency ω_k.

By analogy with the above, the algorithms with the FGD equilibrium and nonequilibrium processing may be formed based on the formula (14). Here, $\gamma(\omega_k)$ is taken along the whole frequency band to be equal to unit for the equilibrium algorithms, and $\gamma(\omega_k)$ is described by formula (10) for nonequilibrium algorithms.

3 Results

To investigate the effectiveness of the proposed algorithms for the case of a weak signal and the correlated sampling PFC, numerical experiments were conducted. For this purpose, a mathematical model of the form (1) was built, where the desired signal on the momentum from the bell tower of the envelope had a duration of 60 ms, and the noise was generated with a random number and normal distribution. PFC mixture of signal and noise was calculated by DFT in the range of 20–60 Hz in steps of 1 Hz sampling. With implementation of algorithms with a nonequilibrium process as a function, the weight function has the form of a triangle.

Figure 1 demonstrates plots of the displacement, and Fig. 2 shows plots of the standard deviation estimates of the temporal state of the signals received by algorithms with equilibrium and nonequilibrium PFC processing depending on the signal-to-noise ratio.

From the analysis results, the accuracy of the estimates of phase response of the method is quite large, so with a signal-to-noise ratio of 0.9, the shift time position estimation is in the range of 1.2–2.2 ms (Fig. 1), and the standard deviation assesses the temporary state of the signal with a signal-to-noise ratio of 1.3 that is less than 17 ms (Fig. 2). In this case, the nonequilibrium processing methods generally provide more accurate estimates in the sense of reducing the standard deviation (Fig. 2). For signal-to-noise ratio less than 1, the movement of the maximum of the weighting function in the higher frequencies leads to a reduction of the standard deviation

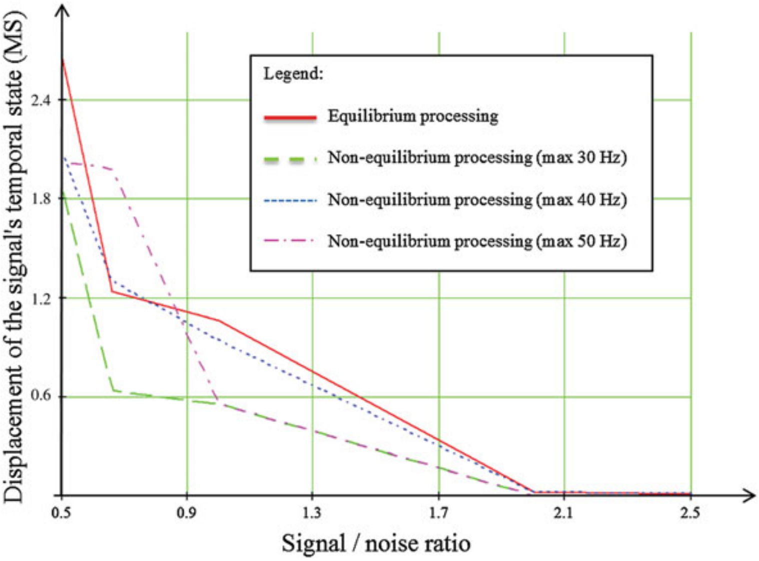


Fig. 1 The dependence of the bias in the temporal state of the signal from the signal-to-noise ratio

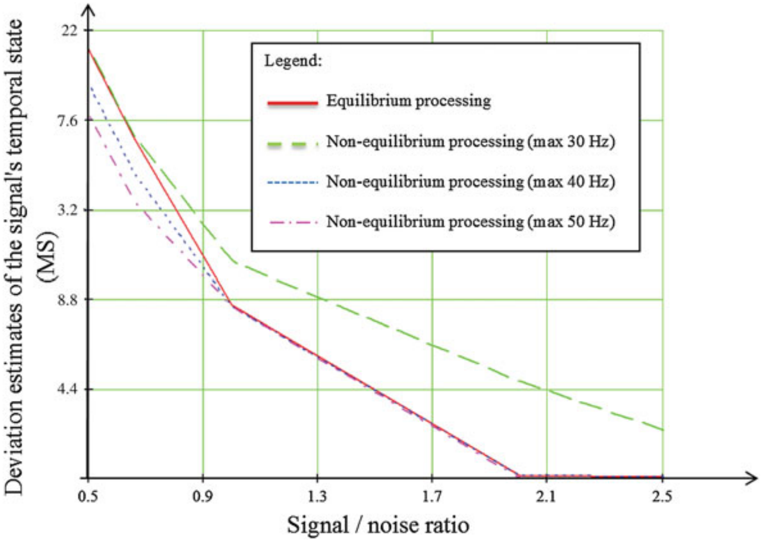


Fig. 2 The dependence of the standard deviation estimates the temporary position of the signal from the signal-to-noise ratio

(Fig. 2), but the minimum value of the bias is already provided with the equilibrium phase frequency methods of treatment (Fig. 1).

A synthetic wavefield with two identical waveforms (Fig. 3a) was used to study the resolution of the nonequilibrium picking algorithm. We used triangular function (10), as a weight, and a bell-shaped wavelet with the central frequencies 24 and 34 Hz.

The nonequilibrium algorithm shows quite a high resolution, of the order of 1/4 wavelength in the vertical dimension (Fig. 3b), and is thus applicable to data distorted by interference [12].

The developed algorithms have been used to process the wave seismic fields resulted in the oil and gas search in the Tomsk region. As an example, Fig. 4 shows the fragments of the original wave field (Fig. 4a) and the field after application of the phase-frequency algorithm with nonequilibrium processing (Fig. 4b). The comparison of the findings shows that the phase-frequency algorithms allowed significant increase in the recording resolution, and thereby the identification of a number of reflection horizons, which are not visible in the original field. At the same time, the temporal position of the identified reflection horizons is well consistent with a priori geological information obtained from geophysical studies of the wells drilled earlier in the given area.

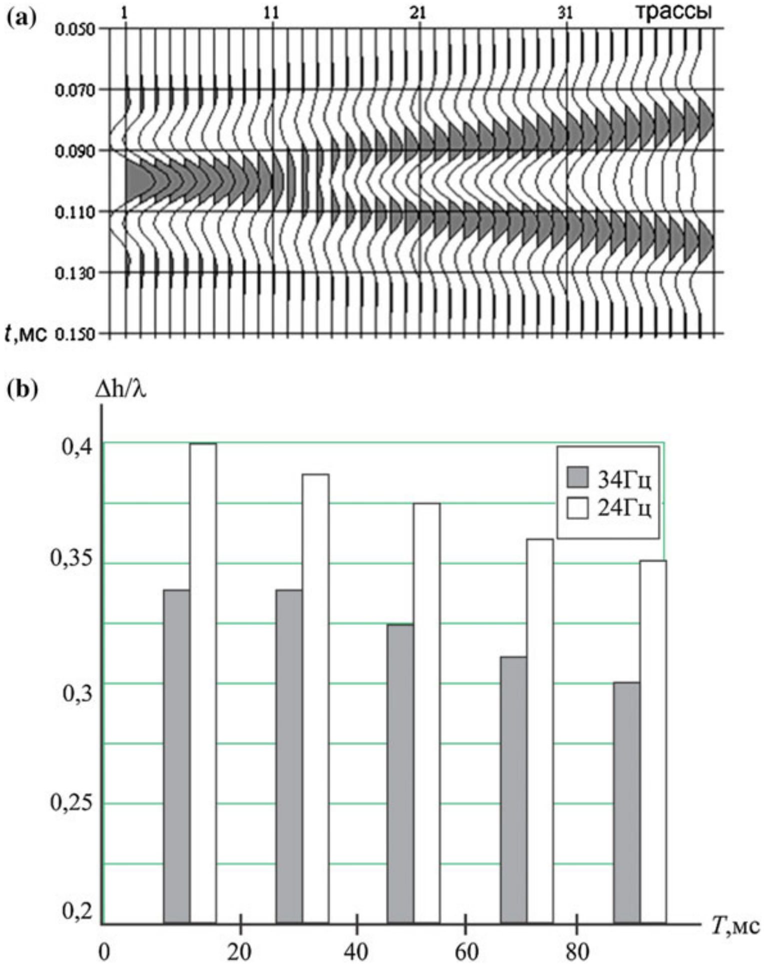


Fig. 3 Resolution of the nonequilibrium phase/frequency picking algorithm. **a** Synthetic seismogram; **b** Respective vertical resolution (Δh), λ is the wavelength

4 Conclusions

To conclude the research, it is necessary to point out that the results of analysis show that the accuracy of the phase frequency method estimates is sufficiently high. At the same time, the nonequilibrium processing methods generally provide more accurate estimates in the sense of reducing the standard deviation.

Thus, checking the accuracy of the study showed that the phase frequency methods with equilibrium and nonequilibrium processing provide sufficient accuracy to obtain time position signal estimates. In addition, their implementation requires just information about the PFC recorded signals. In the future, we plan to study the

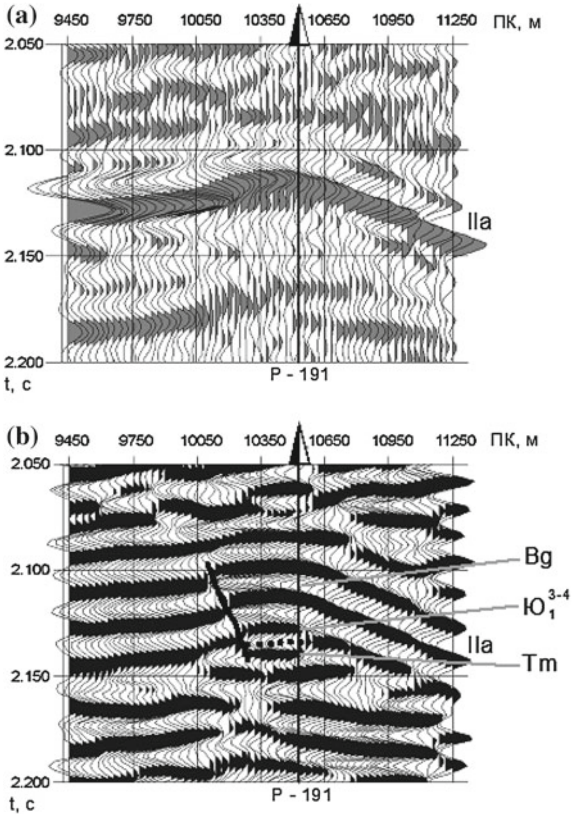


Fig. 4 Fragments of the wave seismic field. **a** The original field; **b** The field after the application of phase-frequency algorithms

accuracy of these methods in the case of calculating PFC with the derivatives of the coefficients of the Fourier series.

The work is performed out of the grant (agreement 14.575.21.0023, project unique identifier 2014-14-576-0052-044) funds in the framework of the Program for Competitive Growth of Tomsk Polytechnic University.

References

1. Oppenheim, A.V., Allen, R.L.: Application of Digital Processing of Signals, 552 pp. Mir, Moscow (1980)
2. Allen, R.L., Mills, D.W.: Signal Analysis: Time, Frequency, Scale, and Structure, 966 p. Wiley-IEEE Press, New York (2004)
3. Oppenheim, A.V., Schafer, R.W., Buck, J.R.: Discrete-Time Signal Processing, 871 p. Prentice-Hall, Publisher (1999)
4. Ivanchenkov, V.P., Kochegurov, A.I.: Estimating arrival times of seismic signals from their phases and frequencies. Geologiya i Geofizika **9**, 77–83 (1988)

5. Ivanchenkov V.P., Kochegurov, A.I.: Phase-frequency algorithms to assess the location of the space-time signals in prior uncertainty. In: Proceedings of Universities. Physics, pp. 100–104 (1995)
6. Khudyakov, G.: On the potential accuracy of determining the position of fluctuating signals. Probl. Electr. Gen. Quest. Electron. **8**, 55–60 (1984)
7. Kochegurov, A.: Analysis of algorithms for measuring the time position of complex signals to evaluate their phase-frequency characteristics. Probl. Inform. **2**(10), 44–50 (2011)
8. Ivanchenkov, V.P., Vylegzhanin, O.N., Orlov, O.V., Kozlov, A.A., Kochegurov, A.I.: Methods, of phase-frequency analysis of wavefields and their application to seismic data processing. Trans. Tomsk Polytech. Univ. **309**(7), 65–70 (2006)
9. Tyapkin, Y.K.: Optimal linear phase bandpass filtering of seismic data. Geologiya i Geofizika **3**, 99–105 (1984)
10. Savarensky, E.F.: Seismic Waves, 296 pp. Nedra, Moscow (1972)
11. Kochegurov, A.I., Bystrov, V.N.: Complex signal time position determination in a medium with dispersion and absorption. Izvestiya Vysshikh Uchebnykh Zavedenij, Radioelektronika **3**, 50–54 (2002)
12. Kochegurov, A.I., Kochegurova, E.A.: Equilibrium and non-equilibrium phase/frequency picking algorithms. Problemy Informatiki **4**(16), 58–64 (2012)

Application of Mini-Models to the Interval Information Granules Processing

Marcin Pluciński

Abstract This paper describes how interval information granules are processed by mini-models. Mini-models are local regression models that can be used for the function approximation learning. In this paper are presented mini-models based on hyperspheres and researches were made for linear and nonlinear models with no limitations on the problem input space dimension. As the output of the model has a form of the interval, a special method of model error calculation was elaborated.

Keywords Mini-model · Local regression · Interval analysis · Data uncertainty · Information granule

1 Introduction

Data which we deal with are very diverse in nature. In most cases, the data are numerical or nominal, thus their processing is well researched, described, and available in a large number of methods. However, information is often imprecise, comprehensible to humans, but difficult to process by computers. People commonly use the so-called information granules—they have the ability to construct and manipulate them. Issues of such type of data processing explore the theory of Granular Computing, which forms a unified conceptual and computing platform. It directly benefits from the already existing and well-established concepts of information granules formed in the setting of set theory, interval arithmetic, fuzzy sets, rough sets, and others [1–7].

It is important that mathematical models processing information granules should have the ability to cope with different (often mixed) types of data. For example, a model created on the basis of crisp data should be able to process information granules, but also we would like to be able to use the granules in the creation of the

M. Pluciński (✉)
Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, Żołnierska 49, 71-062 Szczecin, Poland
e-mail: mplucinski@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_4

model. In this paper, information granules will have the form of multidimensional intervals. For their processing, will be used interval arithmetic [3] and memory-based models.

Memory-based learning methods are an attractive approach compared to creating global models based on a parametric representation. In some situations (for example, small number of samples), building of global models can be difficult and memory-based methods then become one of the possible solutions for the approximation task.

Memory-based methods are well explored and described in many bibliography positions. One of the simplest, for example, is the k-nearest neighbors method (kNN) [8–11]. Another approach can be methods based on a locally weighted learning [8, 12]. Methods widely applied in this category are probabilistic neural networks and generalised regression networks [13, 14].

The concept of mini-models was introduced by Prof. Andrzej Piegat. In papers [15, 16] were described local regression models based on simplexes. The described models were linear and research work was conducted only for problems in one- and two-dimensional input space. In this paper, there are applied mini-models based on hyperspheres described in [17, 18]. Such mini-models can be linear or nonlinear and have no limitations for the problem input space dimension.

2 Linear and Nonlinear Mini-models

The mini-model is a local regression and the answer for the question point $\mathbf{x}^*$ is calculated on the basis of a local model created for k-nearest neighbours. The mini-model is always created when answering calculations (this means that as in other memory method samples are only memorised during learning and the real mini-model is created only when the output is calculated for the given question point $\mathbf{x}^*$) [17, 18]).

In the simplest case the linear mini-model can be applied, and then the answer is calculated on the basis of the linear regression:

$$f(\mathbf{x}^*) = \mathbf{w}^T \cdot \mathbf{x}^*, \tag{1}$$

where: $\mathbf{w}$—the vector of linear mini-model coefficients found for k-neighbours.

In papers [15, 16] are described mini-models created for sectors of the input space that have a triangle shape (in a two-dimensional input space) or a simplex shape in a multidimensional input space. Such sector will be called a mini-model base. In this paper, mini-models will be created for a circular base in a two-dimensional input space or a spherical (hyperspherical) base in a three- or multidimensional input space.

The mini-model base has a centre in the question point $\mathbf{x}^*$ and its radius is defined by a distance between the point $\mathbf{x}^*$ and the most distant point from k neighbours, Fig. 1.

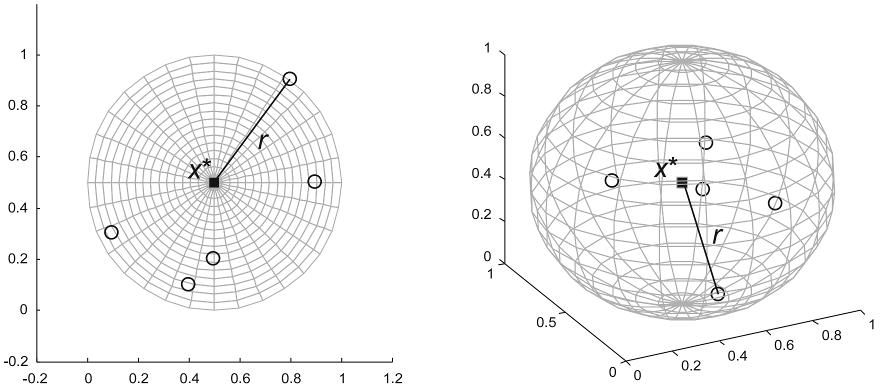


Fig. 1 The mini-model base for a two- and three-dimensional input space

Nonlinear mini-models have better possibilities of fitting to samples. An answer for such model is the sum of a linear mini-model and an additional nonlinear component:

$$f(\mathbf{x}^*) = \mathbf{w}^T \cdot \mathbf{x}^* + f_N(\mathbf{x}^*). \tag{2}$$

As the mini-model is usually created for a small number k of nearest neighbours, the nonlinear function f_N should have a possibility of changing its shape, thanks to as small number of coefficients as possible (because $N + 1$ coefficients must be tuned in the vector $\mathbf{w}$ in an N-dimensional input space).

Among many inspected functions, very advantageous properties have the function:

$$f_N(\mathbf{x}) = w_N \cdot \sin\left[\frac{\pi}{2} - ||\mathbf{x} - \mathbf{x}^*||\frac{\pi}{r}\right], \tag{3}$$

where r is the radius of the mini-model base. In such created function we have only one coefficient w_N to learn. During learning we must find such an w_N value to obtain the best fit of the mini-model to k neighbours. Exemplary linear and nonlinear mini-models in one- and two-dimensional input space are presented in Fig. 2.

One of the main mini-model parameters is the number of neighbours k that are taken into account. It can be constant for the entire data set, but in some approaches it can be dynamically varied—according to the question point location in the input space. The popular technique of k evaluation is applying 'leave one out' cross-validation or applying two distinct data sets: training data—which is memorised, and testing data—to evaluate the real model error. The best k value is the value that gives the lowest test or cross-validation error. The lowest test or cross-validation error guarantees the lowest real error of the model and the best generalisation [18].

The approximation function based on mini-models has many useful properties [17]. It has good accuracy and very advantageous extrapolation features. Learning of the approximator is fast and not computationally complex.

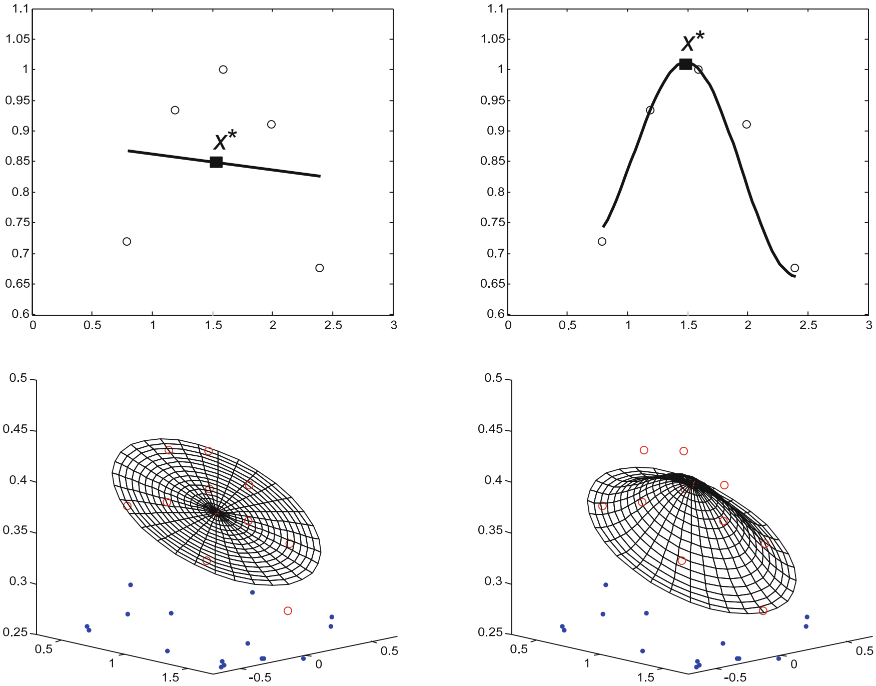


Fig. 2 Exemplary linear and nonlinear mini-models in one- and two-dimensional input space

3 Processing of Interval Information Granules

3.1 Interval Analysis

In this paper will be analysed data that were made uncertain by applying an interval expansion of size α in all dimensions of the data set (both in input and output attributes) [3, 18]. After expansion each attribute becomes the interval number and can be defined as an ordered pair of real numbers $[a, b]$ with $a < b$ such that [3]:

$$[a, b] = \{x : a \leq x \leq b\}. \tag{4}$$

During expansion each crisp attribute x is replaced by the interval $[\underline{x}, \overline{x}]$ where $\underline{x}$ represents the lower interval bound and $\overline{x}$ the upper interval bound:

$$[\underline{x}, \overline{x}] = [x - \alpha, x + \alpha]. \tag{5}$$

The special interval number arithmetic is described in detail in [3]. With its application it is possible to apply mini-models after some small modifications. First of all, the question point $\mathbf{x}^*$ and memorised data can be interval values now, so the method of calculation of the distance between them must be determined to find k neighbours.

The distance $\mathrm{dist}(v_1, v_2)$ between intervals v_1 and v_2 can be calculated in many ways. It is important that the function $\mathrm{dist}(v_1, v_2)$ must fulfil all properties of a metric, so: $\mathrm{dist}(v_1, v_2) = 0$ if and only if $v_1 = v_2$, $\mathrm{dist}(v_1, v_2) = \mathrm{dist}(v_2, v_1)$ and $\mathrm{dist}(v_1, v_2) \leq \mathrm{dist}(v_1, v_3) + \mathrm{dist}(v_3, v_2)$ for any v_3. It can be easily proved that all these properties are fulfilled by the function:

$$\mathrm{dist}(v_1, v_2) = \max\{|\underline{v_1} - \underline{v_2}|, |\overline{v_1} - \overline{v_2}|\}.$$

Finally, the distance between samples can be calculated as follows:

$$\mathrm{dist}(\mathbf{x}, \mathbf{x}^*) = \left(\sum_{j=1}^{N} \mathrm{dist}^2(x_j, x_j^*) \right)^{1/2},$$

where x_j is the jth attribute of the vector $\mathbf{x}$.

Additionally, an inverse matrix to a matrix of intervals cannot be calculated for reason of interval arithmetic limitations. Therefore, the vector $\mathbf{w}$ from the linear regression (1) is calculated for midpoints of intervals which are k neighbours of the question point $\mathbf{x}^*$. Similarly, calculations of the optimal value of the w_N coefficient from the formula (3) are also performed for midpoints of chosen samples.

In experiments described in Sect. 4, the Matlab toolbox INTLAB (created by S.M. Rump and described in [19]) was applied for calculations on interval numbers. Mini-models created with the application of the interval arithmetic work correctly both for interval data and crisp data (Fig. 3).

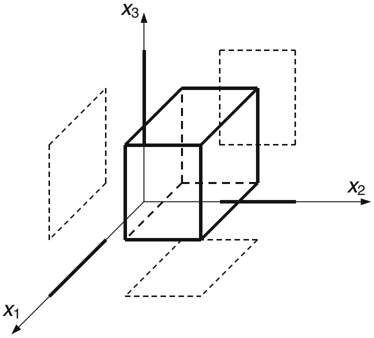


Fig. 3 Exemplary interval information granule in three-dimensional space

3.2 Accuracy of the Model

Let us assume that we have two normalised distinct data sets. The first of them contains training data (although in the case of memory methods like the mini-models method there is no real training process) and each data sample consists of input vector $\mathbf{x}_k$ and target output value y_k, $k = 1 \ldots L$. The second data set contains testing data and each data sample also consists of input vector $\mathbf{x}_i$ and target output value y_i, $i = 1 \ldots M$.

For the crisp data, an error of modelling for the single testing sample $\mathbf{x}_i$ can be evaluated exemplary as:

$$\delta_i = |y_i - y_i^*|, \tag{6}$$

where: y_i^* is the model answer for the question point $\mathbf{x}_i$. A mean absolute error for the entire testing data set can be calculated as:

$$e_{\mathrm{MAE}} = \frac{1}{M} \sum_{i=1}^{M} |y_i - y_i^*|. \tag{7}$$

Now, we must take into account that data are uncertain so an accuracy of the model should be evaluated in a different way. The data are interval values so the accuracy will be also the interval value.

Both y_i and y_i^* will be interval numbers so their subtraction result can be evaluated as:

$$d_i = y_i - y_i^* = \left[\underline{y_i} - \overline{y_i^*}, \overline{y_i} - \underline{y_i^*}\right],$$

according to interval arithmetic [3]. The model error will be also interval number:

$$\delta_i = [\underline{\delta_i}, \overline{\delta_i}], \tag{8}$$

where the lower bound can be calculated as:

$$\underline{\delta_i} = \begin{cases} 0 & \text{if } \; y_i \cap y_i^* \neq \emptyset \\ \min\{|\underline{d_i}|, |\overline{d_i}|\} & \text{otherwise,} \end{cases} \tag{9}$$

and the upper bound as:

$$\overline{\delta_i} = \max\{|\underline{d_i}|, |\overline{d_i}|\}. \tag{10}$$

Figure 4 illustrates the way of calculating the lower and upper bounds of the model error.

Now we can evaluate an interval value for the mean absolute error of the entire testing set as:

$$e_{\mathrm{MAE}} = \left[\underline{e_{\mathrm{MAE}}}, \overline{e_{\mathrm{MAE}}}\right],$$

where:

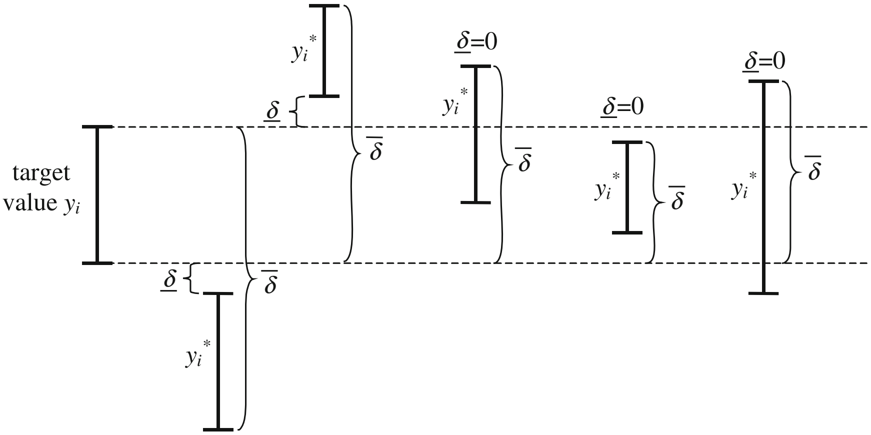


Fig. 4 Exemplary calculations of the lower and upper bounds of the model error

$$\underline{e_{\text{MAE}}} = \frac{1}{M}\sum_{i=1}^{M}\underline{\delta_i} \quad \text{and} \quad \overline{e_{\text{MAE}}} = \frac{1}{M}\sum_{i=1}^{M}\overline{\delta_i}.$$

Next, additionally let us define the notion of the model accuracy as:

$$q = \frac{1}{1 + e_{\text{MAE}}} = [\underline{q}, \overline{q}] = \left[\frac{1}{1 + \overline{e_{\text{MAE}}}}, \frac{1}{1 + \underline{e_{\text{MAE}}}}\right]. \tag{11}$$

The accuracy defined in such way changes its value in the range from 0 (for model errors approaching infinity) to 1 (for model errors approaching 0). It is clear that always $\underline{q} \leq \overline{q}$.

In a similar way, a cross-validation error and the cross-validation accuracy can be calculated.

4 Experiments

The main task of preliminary experiments was an examination if interval data are correctly processed by mini-models.

In the first experiment, one-dimensional data was used for model creation. The data had a form of noisy sine function (100 samples), Fig. 5a. The crisp data were submitted to the interval expansion of the size α (both input and output) in the way described in Sect 3.1, Fig. 5b. In experiments in which data were crisp ($\alpha = 0$), they were also treated as intervals but in a degenerated form (i.e., $\underline{x} = \overline{x}$). Thanks to this, the main rule for the model work was the same for both crisp and interval data.

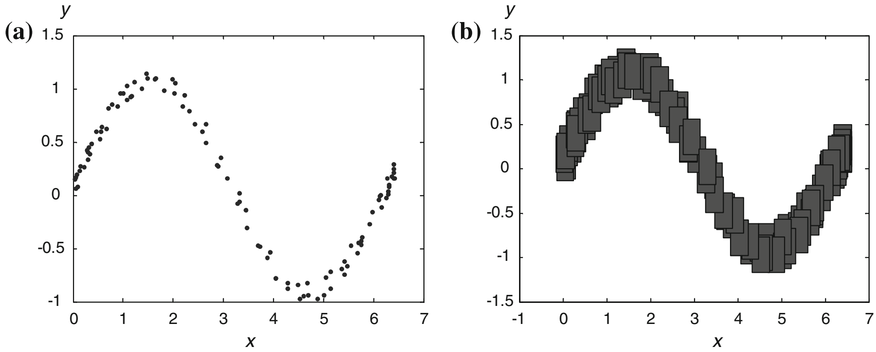


Fig. 5 One-dimensional data used in preliminary experiments

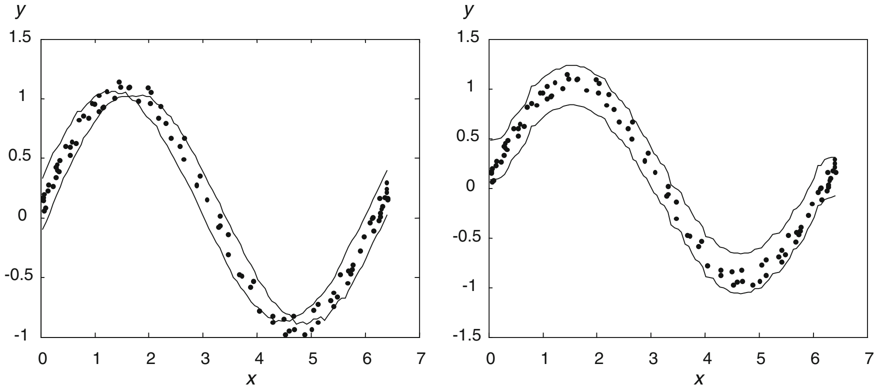


Fig. 6 Lower and upper bounds of the output interval of the linear mini-model (on the *left*) and of the weighted kNN method (on the *right*)

Figure 6 presents plots of the lower and upper bounds of the output interval of the linear mini-model found for an input interval (with width equal 0.2) taking values from 0 to 2π. Learning data of the model (visible on the plot as points) were also expanded to intervals with width equal 0.2 and the number of nearest neighbours taken as 15. For comparison, on the right we can see the plot created for weighted kNN method.

Figures 7 and 8 presents a cross-validation error of the model created for data also expanded to intervals as before. In Fig. 7 we can see a variation of the lower and upper bounds of the cross-validation error interval for different number of neighbours k taken into account. In experiment $k \in (3, 20)$, $\alpha = 0.03$ and the mini-model was linear. We can find from the figure that the most advantageous k value is equal to 7. It guarantees the lowest cross-validation error and thereby the best generalisation.

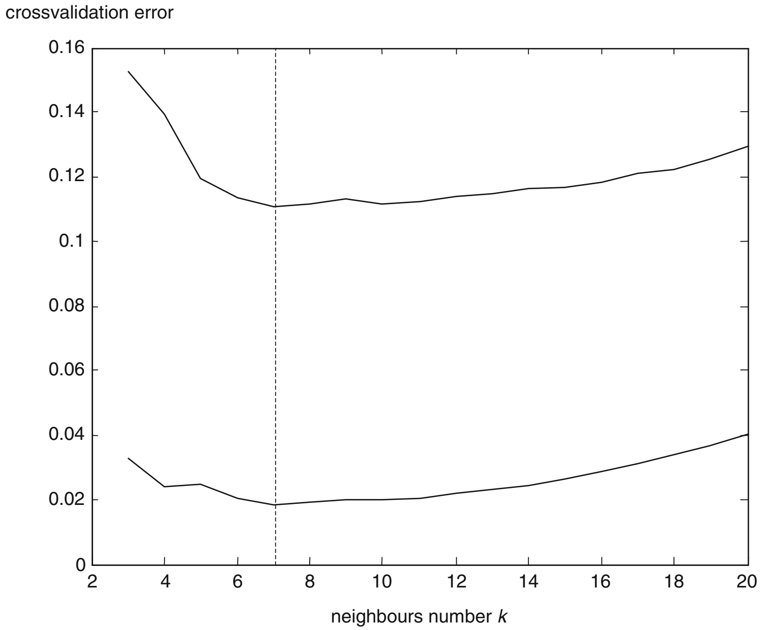


Fig. 7 Lower and upper bounds of the cross-validation error interval for different numbers of neighbours k

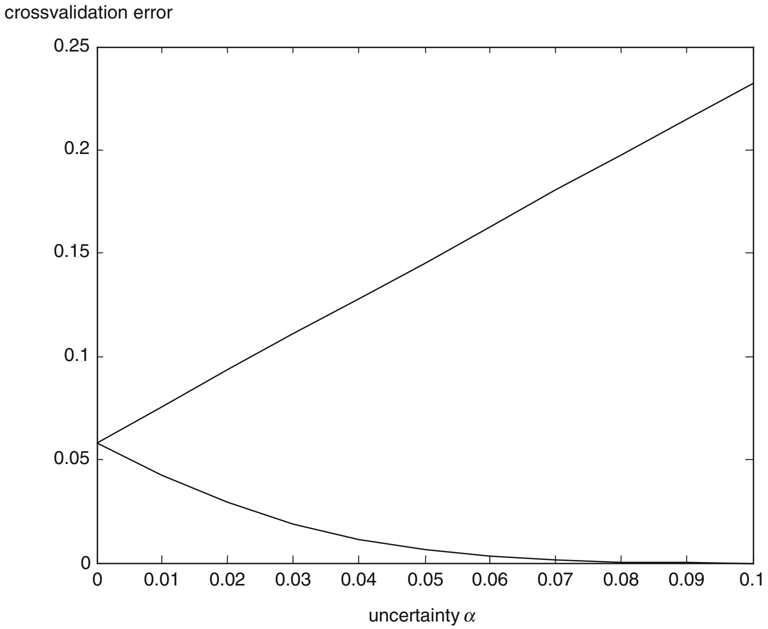


Fig. 8 Lower and upper bounds of the cross-validation error interval for interval width α taking values from 0 to 0.1

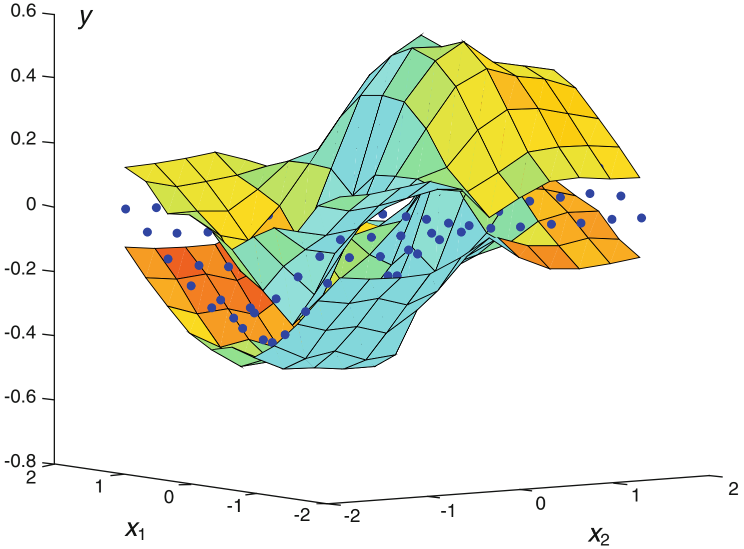


Fig. 9 Lower and upper bounds of the output interval found for model with two inputs

Figure 8 presents the lower and upper bounds of the cross-validation error interval for the best number of neighbours $k = 7$ and the interval width α (of learning data) taking values from 0 to 0.1.

Figure 9 presents the lower and upper bounds of the output interval found for model with two inputs. Learning data was generated on the basis of function:

$$y = \frac{\sin(x_1) \cdot \cos(x_2)}{x_1^2 + x_2^2 + 1} \tag{12}$$

for $x_1, x_2 \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ and sampling step equal to 0.25. In this experiment, learning data was crisp and the input vector was expanded to interval with the width $\alpha = 0.5$.

In the second part of the experiments was calculated a real accuracy of function approximators based on mini-models, Table 1. The research work was performed on data created by the author and data from popular Web repositories. Data were normalised due to different ranges of its inputs and submitted to the interval expansion of size $\alpha = 0.01$.

The real error was calculated with an application of the left out cross-validation method. Mini-models are compared with kNN method and in each case calculation results are presented for an optimal number of neighbours (giving the lowest real error). In each table cell there are two numbers: lower and upper bound of the error interval.

Table 1 The real error of function approximators

Data	Inputs number	Mean kNN	Weighted mean kNN	Linear mini-model	Nonlinear mini-model
$\sin(x)$ with 0.5 sampling step	1	0.065	0.063	0.025	0.024
		0.104	0.102	0.076	0.069
$\sin(x)$ with 0.2 sampling step	1	0.008	0.008	0	0
		0.035	0.035	0.036	0.033
$\frac{\sin(x_1)*\cos(x_2)}{x_1^2+x_2^2+1}$ with 0.25 sampling step	2	0.018	0.017	0.015	0.016
		0.052	0.052	0.045	0.048
$\frac{\sin(x_1)*\cos(x_2)}{x_1^2+x_2^2+1}$ with 0.1 sampling step	2	0.001	0.001	0.0004	0.0005
		0.027	0.027	0.028	0.029
`bodyfat`	14	0.030	0.030	0.003	0.003
		0.067	0.067	0.036	0.036
`cpu`	6	0.014	0.014	0.010	0.010
		0.046	0.045	0.046	0.047
`diabetes_numeric`	2	0.113	0.114	0.117	0.120
		0.152	0.154	0.148	0.151
`elusage`	2	0.080	0.081	0.075	0.080
		0.117	0.118	0.119	0.122

5 Conclusions

First of all, the approximation function base on mini-models proved to have good accuracy, Table 1. The accuracy is particularly great for data without noise. In the case of noised data, mini-models have accuracy comparable or slightly worse than kNN methods. As the output of the model has a form of the interval, there was a necessity for elaboration of a special method of model error calculation.

Learning of the approximator is very fast—it is enough to memorise learning data and the proper mini-model is created only when calculating an answer for the question point $\mathbf{X}^*$. Mini-models creating is not computationally complex because they are built on the base of a small number of samples (k nearest neighbours). The linear mini-model is a linear regression found for k neighbours and the nonlinear one has only one additional coefficient to compute.

All experiments show that models created on the base of mini-models work correctly, processing of intervals is fast and carry on without any problems.

References

1. Duch, W.: Uncertainty of data, fuzzy membership functions, and multilayer perceptrons. IEEE Trans. Neural Netw. **16**(1), 1023 (2005)
2. Gacek, A.: Granular modelling of signals: a framework of granular computing. Inf. Sci. **221**, 1–11 (2013)
3. Moore, R.E., Kearfott, R.B., Cloud, M.J.: Introduction to Interval Analysis. Society for Industrial and Applied Mathematics. SIAM, Philadelphia (2009)
4. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. Springer (1991)
5. Pedrycz, W., Skowron, A., Kreinovich, V. (eds.): Handbook of Granular Computing. Wiley, Chichester (2008)
6. Piegat, A.: Fuzzy Modeling and Control. Physica Verlag, Heidelberg (2001)
7. Zadeh, L.A.: Towards a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. Fuzzy Sets Syst. **90**, 111–117 (1997)
8. Cichosz, P.: Learning Systems. WNT Publishing House, Warsaw (2000) [in Polish]
9. Hand, D., Mannila, H., Smyth, P.: Principles of Data Mining. The MIT Press (2001)
10. Kordos, M., Blachnik, M., Strzempa, D.: Do we need whatever more than k-NN? In: Rutkowski, L., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2010. LNCS, vol. 6113, pp. 414–421. Springer, Heidelberg (2010)
11. Moore, A.W., Atkeson, C.G., Schaal, S.A.: Memory-based learning for control. Technical Report CMU-RI-TR-95-18, Carnegie-Mellon University, Robotics Institute (1995)
12. Atkeson, C.G., Moore, A.W., Schaal, S.A.: Locally weighted learning. Artif. Intell. Rev. **11**, 11–73 (1997)
13. Wasserman, P.D.: Advanced Methods in Neural Computing. Van Nostrand Reinhold, New York (1993)
14. Wright, W.A.: Bayesian approach to neural network modeling with input uncertainty. IEEE Trans. Neural Netw. **10**(6), 1261–1270 (1999)
15. Piegat, A., Wąsikowska, B., Korzeń, M.: Application of the self-learning, 3-point mini-model for modeling of unemployment rate in Poland. Studia Informatica, University of Szczecin (2010) [in Polish]
16. Piegat, A., Wąsikowska, B., Korzeń, M.: Differences between the method of mini-models and the *k*-nearest neighbors on example of modeling of unemployment rate in Poland. In: Proceedings of 9th Conference on Information Systems in Management, pp. 34–43. WULS Press, Warsaw (2011)
17. Pluciński, M.: Mini-models—local regression models for the function approximation learning. In: Rutkowski, L., et al. (eds.) ICAISC 2012, Part II. LNCS, pp. 160–167. Springer, Heidelberg (2012)
18. Pluciński, M.: Evaluation of the mini-models robustness to data uncertainty with the application of the information-gap theory. In: Rutkowski, L., et al. (eds.) ICAISC 2013, Part II. LNAI, pp. 230–241. Springer, Heidelberg (2013)
19. Rump, S.M.: INTLAB—INTerval LABoratory. Developements in Reliable Computing, pp. 77–104. Kluwer Academic Publishing, Dordrecht (1999)

A Parametric Interval Approximation of Fuzzy Numbers

Luca Anzilli and Gisella Facchinetti

Abstract In this paper we present a parametric formulation of interval approximation of fuzzy numbers. It is based on a more complex version of generalized Trutschnig et al. distance. General conclusions are showed and particular cases are studied in details.

Keywords Fuzzy sets · Fuzzy quantities · Interval type-2 fuzzy sets · Evaluation fuzzy numbers · Interval approximation · Weighting functions

1 Introduction

The problem to approximate a fuzzy set has been studied by several authors. Some of them use an interval approximation [1, 4–11], others use triangular or trapezoidal approximation. This idea is born with the aim of simplifying complicated calculations that appear in modeling and processing fuzzy optimization and control problems. In all these cases, even if we start with data described by fuzzy numbers of the easier forms like triangular and trapezoidal, the fuzzy outputs we meet are complicated and may have lost all the peculiarity of the starting sets. It is sufficient to look to a usual fuzzy control system output that may be neither normal nor convex.

In this paper we propose to substitute a given fuzzy number with an interval which has some properties like to be the nearest in some sense we describe. The results we present start from a paper of Grzegorzewski [10] in which he proposes the nearest interval to the original fuzzy number with respect to several distances. In particular we have focused on the distance introduced by Trutschnig et al. [13]. This distance depends on two parameters, a constant θ and a function $f(\alpha)$ that depends on the

L. Anzilli · G. Facchinetti (✉)
Department of Management, Economics, Mathematics and Statistics,
University of Salento, Lecce, Italy
e-mail: gisella.facchinetti@unisalento.it

L. Anzilli
e-mail: luca.anzilli@unisalento.it

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_5

variable that individualizes the α-cuts of a fuzzy set. We propose to consider even θ as a function of α. This simple variation generates interesting results and connection with other important intervals connected with the initial fuzzy number defined in previous researches.

In Sect. 2 we give basic definitions and notations. In Sect. 3 we present an interval approximation for a fuzzy number obtained by minimizing a suitable functional. In Sects. 4 and 5 we study some properties of the approximation interval we have proposed. Finally, in Sect. 6 we apply our method to the interval approximation of trapezoidal fuzzy numbers.

2 Preliminaries and Notation

Let X denote a universe of discourse. A fuzzy set A in X is defined by a membership function $\mu_A : X \to [0, 1]$ which assigns to each element of X, a grade of membership to the set A. The height of A is $h_A = \mathit{height}\, A = \sup_{x \in X} \mu_A(x)$. The support and the core of A are defined, respectively, as the crisp sets $\mathit{supp}(A) = \{x \in X;\, \mu_A(x) > 0\}$ and $\mathit{core}(A) = \{x \in X;\, \mu_A(x) = 1\}$. A fuzzy set A is normal if its core is nonempty. A fuzzy number A is a fuzzy set of the real line $\mathbb{R}$ with a normal, fuzzy convex and upper-semicontinuous membership function of bounded support (see, e.g., [2]). In accordance with the definition given above there exist four numbers $a_1, a_2, a_3, a_4 \in \mathbb{R}$, with $a_1 \le a_2 \le a_3 \le a_4$, and two functions $l_A, r_A : \mathbb{R} \to [0, 1]$ called the left side and the right side of A, respectively, where l_A is nondecreasing and right-continuous and r_A is nonincreasing and left-continuous, such that

$$\mu_A(x) = \begin{cases} 0 & x < a_1 \\ l_A(x) & a_1 \le x < a_2 \\ 1 & a_2 \le x \le a_3 \\ r_A(x) & a_3 < x \le a_4 \\ 0 & x > a_4\,. \end{cases}$$

The α-cut of a fuzzy set A, $0 \le \alpha \le 1$, is defined as the crisp set $A_\alpha = \{x \in X;\, \mu_A(x) \ge \alpha\}$ if $0 < \alpha \le 1$ and as the closure of the support if $\alpha = 0$. Every α-cut of a fuzzy number is a closed interval $A_\alpha = [a_L(\alpha), a_R(\alpha)]$, for $0 \le \alpha \le 1$, where $a_L(\alpha) = \inf A_\alpha$ and $a_R(\alpha) = \sup A_\alpha$.

In the following we will employ the mid-spread representation of intervals. The middle point and the spread of the interval $I = [a, b]$ will be denoted, respectively, by $\mathit{mid}(I) = (a + b)/2$ and $\mathit{spr}(I) = (b - a)/2$.

3 Interval Approximation of Fuzzy Numbers

Our proposal starts from the Grzegorzewski papers in which the author defines and finds an interval approximation of a fuzzy number. Starting from a distance between two fuzzy numbers and observing that any closed interval is a fuzzy number, the author defines the approximating interval of a fuzzy number as the interval of minimum distance. The distance he uses is based on the distance between two closed intervals I and J introduced by Trutschnig et al. [13] defined by

$$d_{\bar{\theta}}(I, J) = \sqrt{(mid(I) - mid(J))^2 + \bar{\theta}(spr(I) - spr(J))^2}$$

where the mid-spread representation of the involved intervals is employed. The parameter $\bar{\theta}\ \in]0, 1]$ indicates the relative importance of the spreads against the mids [10, 13]. The distance $d_{\bar{\theta}}$ is extended to the space of all fuzzy number $\mathbb{F}(\mathbb{R})$ by defining $D_{\bar{\theta}} : \mathbb{F}(\mathbb{R}) \times \mathbb{F}(\mathbb{R}) \to [0, +\infty[$ such that for two arbitrary fuzzy numbers A and B

$$D^2_{f,\bar{\theta}}(A, B) = \frac{1}{\int_0^1 f(\alpha)\, d\alpha} \int_0^1 d^2_{\bar{\theta}}(A_\alpha, B_\alpha)\, f(\alpha)\, d\alpha$$

where the weighting function $f\ :]0, 1] \to [0, +\infty[$ is such that $\int_0^1 f(\alpha)\, d\alpha > 0$.

In this paper we extend Grzegorzewski's idea by considering the parameter θ as a function of α having in mind that the relative importance of the spreads against the mids may depend on the level of uncertainty. This hypothesis leads to interesting results.

Definition 1 We say that $C^*(A) = [c^*_L, c^*_R]$ is an approximation interval of the fuzzy number A with respect to the pair (f, θ) if it minimizes the weighted mean of the squared distances

$$\begin{aligned}\mathcal{D}^{(2)}_{f,\theta}(C; A) &= \frac{1}{\int_0^1 f(\alpha)\, d\alpha} \int_0^1 d^2_{\theta(\alpha)}(C, A_\alpha)\, f(\alpha)\, d\alpha \\ &= \frac{1}{\int_0^1 f(\alpha)\, d\alpha} \int_0^1 \left[(mid(C) - mid(A_\alpha))^2 + \theta(\alpha)\, (spr(C) - spr(A_\alpha))^2\right] \\ &\qquad \times f(\alpha) d\alpha\end{aligned}$$

among all the intervals $C = [c_L, c_R]$, where the weight function $f : [0, 1] \to [0, +\infty[$ is such that $\int_0^1 f(\alpha)\, d\alpha > 0$ and $\theta : [0, 1] \to]0, 1]$ is a function that indicates the relative importance of the spreads against the mids [10, 13].

Theorem 1 *The approximation interval $C^*(A) = C^*(A; f, \theta) = [c_L^*, c_R^*]$ of the fuzzy number A with respect to (f, θ) is given by*

$$\begin{aligned} c_L^* &= \frac{\int_0^1 mid(A_\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha} - \frac{\int_0^1 spr(A_\alpha)\, f(\alpha)\, \theta(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, \theta(\alpha)\, d\alpha} \\ c_R^* &= \frac{\int_0^1 mid(A_\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha} + \frac{\int_0^1 spr(A_\alpha)\, f(\alpha)\, \theta(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, \theta(\alpha)\, d\alpha}\,. \end{aligned} \tag{1}$$

Proof We have to minimize the function

$$\begin{aligned} g(c_L, c_R) = &\int_0^1 \left(\frac{c_L + c_R}{2} - \frac{a_L(\alpha) + a_R(\alpha)}{2} \right)^2 f(\alpha) d\alpha \\ &+ \int_0^1 \theta(\alpha) \left(\frac{c_R - c_L}{2} - \frac{a_R(\alpha) - a_L(\alpha)}{2} \right)^2 f(\alpha) d\alpha \end{aligned}$$

with respect to c_L and c_R. We obtain

$$\begin{aligned} \frac{\partial g}{\partial c_L}(c_L, c_R) &= \int_0^1 (mid(C) - mid(A_\alpha))\ f(\alpha) d\alpha \\ &\quad - \int_0^1 \theta(\alpha)\,(spr(C) - spr(A_\alpha))\ f(\alpha) d\alpha \\ \frac{\partial g}{\partial c_R}(c_L, c_R) &= \int_0^1 (mid(C) - mid(A_\alpha))\ f(\alpha) d\alpha \\ &\quad + \int_0^1 \theta(\alpha)\,(spr(C) - spr(A_\alpha))\ f(\alpha) d\alpha\,. \end{aligned}$$

By solving

$$\begin{cases} \frac{\partial g}{\partial c_L}(c_L, c_R) = 0 \\ \frac{\partial g}{\partial c_R}(c_L, c_R) = 0 \end{cases}$$

we obtain that the solution $C^* = C^*(A) = [c_L^*, c_R^*]$ satisfies

$$mid(C^*) = \frac{\int_0^1 mid(A_\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha}\,, \qquad spr(C^*) = \frac{\int_0^1 spr(A_\alpha)\, f(\alpha)\, \theta(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, \theta(\alpha)\, d\alpha}$$

and thus, since $c_L^* = mid(C^*) - spr(C^*)$ and $c_R^* = mid(C^*) + spr(C^*)$, we obtain (1). Moreover, by calculation, we get

$$\frac{\partial^2 g}{\partial c_L^2}(c_L, c_R) = \frac{\partial^2 g}{\partial c_R^2}(c_L, c_R) = \frac{1}{2}\left(\int_0^1 f(\alpha)\, d\alpha + \int_0^1 f(\alpha)\, \theta(\alpha)\, d\alpha \right)$$

and

$$\frac{\partial^2 g}{\partial c_R \partial c_L}(c_L, c_R) = \frac{\partial^2 g}{\partial c_L \partial c_R}(c_L, c_R) = \frac{1}{2}\left(\int_0^1 f(\alpha)\, d\alpha - \int_0^1 f(\alpha)\,\theta(\alpha)\, d\alpha\right)$$

and thus

$$\det\begin{bmatrix} \frac{\partial^2 g}{\partial c_L^2}(c_L, c_R) & \frac{\partial^2 g}{\partial c_R \partial c_L}(c_L, c_R) \\ \frac{\partial^2 g}{\partial c_L \partial c_R}(c_L, c_R) & \frac{\partial^2 g}{\partial c_R^2}(c_L, c_R) \end{bmatrix} = \left(\int_0^1 f(\alpha)\, d\alpha\right)\left(\int_0^1 f(\alpha)\,\theta(\alpha)\, d\alpha\right) > 0$$

and $\frac{\partial^2 g}{\partial c_L^2}(c_L, c_R) > 0$. Then (c_L^*, c_R^*) minimizes $g(c_L, c_R)$.

Remark 1 The previous theorem still holds if $\theta > 0$ almost everywhere in [0, 1].

4 Properties

In this section we study some properties of the approximation interval.

Proposition 1 *The approximation interval*

$$C^*(A) = C^*(A;\, f, \theta) = [c_L^*(A;\, f, \theta), c_R^*(A;\, f, \theta)]$$

given by (1) *satisfies the following properties:*

(i) *invariance under translations, that is*

$$C^*(A + z;\, f, \theta) = C^*(A;\, f, \theta) + z \qquad \forall z \in \mathbb{R}\,;$$

(ii) *scale invariance, that is*

$$C^*(z \cdot A;\, f, \theta) = z \cdot C^*(A;\, f, \theta) \qquad \forall\, z \in \mathbb{R}\backslash\{0\}\,.$$

Proof Let us prove (i). Since

$$mid((A + z)_\alpha) = \frac{1}{2}(a_L(\alpha) + z + a_R(\alpha) + z) = mid(A_\alpha) + z$$

and

$$spr((A + z)_\alpha) = \frac{1}{2}(a_R(\alpha) + z - a_L(\alpha) - z) = spr(A_\alpha)$$

from (1) we obtain

$$\begin{aligned} c_L^*(A+z;\, f,\theta) &= \frac{\int_0^1 mid((A+z)_\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha} - \frac{\int_0^1 spr((A+z)_\alpha)\, f(\alpha)\, \theta(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, \theta(\alpha)\, d\alpha} \\ &= \frac{\int_0^1 mid(A_i^\alpha)\, p_i(\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha} - \frac{\int_0^1 spr(A_i^\alpha)\, f(\alpha)\, \theta(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, \theta(\alpha)\, d\alpha} + z \\ &= c_L^*(A;\, f,\theta) + z\,. \end{aligned}$$

In a similar way, we get $c_R^*(A+z;\, f,\theta) = c_R^*(A;\, f,\theta) + z$ and thus $C^*(A+z;\, f,\theta) = C^*(A;\, f,\theta) + z$.
Let us prove (ii). First, we consider the case $z > 0$. We have

$$(z \cdot A)_\alpha = [z \cdot a_L(\alpha), z \cdot a_R(\alpha)]$$

and thus

$$mid((z \cdot A)_\alpha) = \frac{1}{2}(z \cdot a_L(\alpha) + z \cdot a_R(\alpha)) = z \cdot mid(A_\alpha)$$

and

$$spr((z \cdot A)_\alpha) = \frac{1}{2}(z \cdot a_R(\alpha) - z \cdot a_L(\alpha)) = z \cdot spr(A_\alpha)\,.$$

So from (1) we get

$$c_L^*(z \cdot A;\, f,\theta) = z \cdot c_L^*(A;\, f,\theta)\,, \qquad c_R^*(z \cdot A;\, f,\theta) = z \cdot c_R^*(A;\, f,\theta)$$

and thus

$$C^*(z \cdot A;\, f,\theta) = z \cdot C^*(A;\, f,\theta)\,.$$

If $z < 0$ we have

$$(z \cdot A)_\alpha = [z \cdot a_R(\alpha), z \cdot a_L(\alpha)]$$

and thus

$$mid((z \cdot A)_\alpha) = z \cdot mid(A_\alpha)\,, \qquad spr((z \cdot A)_\alpha) = (-z) \cdot spr(A_\alpha)\,.$$

Then from (1) we obtain

$$\begin{aligned} c_L^*(z \cdot A;\, f,\theta) &= \frac{\int_0^1 mid((z \cdot A)_\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha} - \frac{\int_0^1 spr((z \cdot A)_\alpha)\, f(\alpha)\, \theta(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, \theta(\alpha)\, d\alpha} \\ &= z\frac{\int_0^1 mid(A_\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha} + z\frac{\int_0^1 spr(A_\alpha)\, f(\alpha)\, \theta(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, \theta(\alpha)\, d\alpha} \\ &= z\, c_R^*(A;\, f,\theta)\,. \end{aligned}$$

In the similar way, we get $c_R^*(z \cdot A;\, f, \theta) = z\, c_L^*(A;\, f, \theta)$. Then, taking into account that $z < 0$, we have

$$\begin{aligned} C^*(z \cdot A;\, f, \theta) &= [c_L^*(z \cdot A;\, f, \theta), c_R^*(z \cdot A;\, f, \theta)] = [z\, c_R^*(A;\, f, \theta), z\, c_L^*(A;\, f, \theta)] \\ &= z \cdot [c_L^*(A;\, f, \theta), c_R^*(A;\, f, \theta)] = z \cdot C^*(A;\, f, \theta)\,. \end{aligned}$$

5 Relation to Expected Interval and Interval-Valued Possibilistic Mean

Some important intervals connected with a fuzzy number are often utilized to have its view. We pertain to the *expected interval* $EI(A)$ of a fuzzy number A, introduced by Dubois and Prade [6] and Heilpern [11]

$$EI(A) = \left[\int_0^1 a_L(\alpha)\, d\alpha,\ \int_0^1 a_R(\alpha)\, d\alpha\right],$$

the *interval-valued possibilistic mean* introduced by Carlsson and Fullér [3]

$$M(A) = \left[2\int_0^1 a_L(\alpha)\alpha\, d\alpha,\ 2\int_0^1 a_R(\alpha)\alpha\, d\alpha\right],$$

and the *f-weighted interval-valued possibilistic mean* proposed by Fullér and Majlender [8] for monotonic increasing weighting functions and by Liu [12] without the monotonic increasing assumption

$$M_f(A) = \left[\frac{\int_0^1 a_L(\alpha) f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha},\ \frac{\int_0^1 a_R(\alpha) f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha}\right]$$

which is a generalization of the previous ones. It is interesting to see that there is an important connection between $M_f(A)$ and the approximation interval $C_{f,\theta}^*(A) = C^*(A;\, f, \theta)$ we have introduced before. As

$$mid(M_f(A)) = \frac{\int_0^1 mid(A_\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha}, \qquad spr(M_f(A)) = \frac{\int_0^1 spr(A_\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha}, \tag{2}$$

and observed that from (1) we have

$$mid(C^*_{f,\theta}(A)) = \frac{\int_0^1 mid(A_\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha}\,,$$

$$spr(C^*_{f,\theta}(A)) = \frac{\int_0^1 spr(A_\alpha)\, f(\alpha)\, \theta(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, \theta(\alpha)\, d\alpha} \tag{3}$$

we get

$$mid(C^*_{f,\theta}(A)) = \frac{\int_0^1 mid(A_\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha} = mid(M_f(A)) \tag{4}$$

and thus $M_f(A)$ and $C^*_{f,\theta}(A)$ have the same middle point independently of the choice of θ. Obviously, they may differ in their spreads. Thus, for a given weighting functions f, they are all intervals centered at the same point but with different sizes. May we say something about these sizes? The following considerations reply to some questions.

To this aim, we consider the preference index value of the weighting function f

$$e_f = \frac{\int_0^1 \alpha\, f(\alpha)\, d\alpha}{\int_0^1 f(\alpha)\, d\alpha} \tag{5}$$

introduced in [12]. Similarly, we define the preference index value of the function $\theta \cdot f$ as

$$k_{f,\theta} = \frac{\int_0^1 \alpha\, \theta(\alpha)\, f(\alpha)\, d\alpha}{\int_0^1 \theta(\alpha)\, f(\alpha)\, d\alpha} \tag{6}$$

and

$$\varepsilon_f(\theta) = e_f - k_{f,\theta}\,. \tag{7}$$

First, we prove the following lemma.

Lemma 1 *Let* $\tilde{f}, \tilde{g} : [0, 1] \to [0, +\infty[$ *such that* $\int_0^1 \tilde{f}(\alpha)\, d\alpha = 1$, $\int_0^1 \tilde{g}(\alpha)\, d\alpha = 1$ *and*

$$\forall \alpha, \gamma \in [0, 1] \qquad \alpha \geq \gamma \implies \tilde{f}(\alpha)\tilde{g}(\gamma) - \tilde{f}(\gamma)\tilde{g}(\alpha) \geq 0\,. \tag{8}$$

Then if h is an increasing function we have

$$\int_0^1 h(\alpha)\tilde{f}(\alpha)\, d\alpha - \int_0^1 h(\alpha)\tilde{g}(\alpha)\, d\alpha \geq 0;$$

if h is a decreasing function we have

$$\int_0^1 h(\alpha)\tilde{f}(\alpha)\, d\alpha - \int_0^1 h(\alpha)\tilde{g}(\alpha)\, d\alpha \leq 0\,.$$

Proof We have

$$
\begin{aligned}
&\int_0^1 h(\alpha)\tilde{f}(\alpha)\,d\alpha - \int_0^1 h(\alpha)\tilde{g}(\alpha)\,d\alpha \\
&= \int_0^1 h(\alpha)\tilde{f}(\alpha)\,d\alpha \int_0^1 \tilde{g}(\gamma)\,d\gamma - \int_0^1 h(\gamma)\tilde{g}(\gamma)\,d\alpha \int_0^1 \tilde{f}(\alpha)\,d\alpha \\
&= \int_0^1 \int_0^1 (h(\alpha) - h(\gamma))\tilde{f}(\alpha)\tilde{g}(\gamma)\,d\alpha\,d\gamma \\
&= \int_0^1 d\alpha \int_0^\alpha (h(\alpha) - h(\gamma))\tilde{f}(\alpha)\tilde{g}(\gamma)\,d\gamma + \int_0^1 d\alpha \int_\alpha^1 (h(\alpha) - h(\gamma))\tilde{f}(\alpha)\tilde{g}(\gamma)\,d\gamma \\
&= \int_0^1 d\alpha \int_0^\alpha (h(\alpha) - h(\gamma))\tilde{f}(\alpha)\tilde{g}(\gamma)\,d\gamma + \int_0^1 d\gamma \int_0^\gamma (h(\alpha) - h(\gamma))\tilde{f}(\alpha)\tilde{g}(\gamma)\,d\alpha \\
&= \int_0^1 d\alpha \int_0^\alpha (h(\alpha) - h(\gamma))\tilde{f}(\alpha)\tilde{g}(\gamma)\,d\gamma + \int_0^1 d\alpha \int_0^\alpha (h(\gamma) - h(\alpha))\tilde{f}(\gamma)\tilde{g}(\alpha)\,d\gamma \\
&= \int_0^1 d\alpha \int_0^\alpha (h(\alpha) - h(\gamma))[\tilde{f}(\alpha)\tilde{g}(\gamma) - \tilde{f}(\gamma)\tilde{g}(\alpha)]\,d\gamma\,.
\end{aligned}
$$

The assertions follow from (8).

Proposition 2 *Let* $\theta_1, \theta_2 : [0,1] \to]0,1]$ *such that*

$$
\forall\,\alpha, \gamma \in [0,1] \qquad \alpha \geq \gamma \implies \theta_1(\alpha)\theta_2(\gamma) - \theta_1(\gamma)\theta_2(\alpha) \geq 0\,. \tag{9}
$$

Then

$$
\varepsilon_f(\theta_2) \geq \varepsilon_f(\theta_1)
$$

and

$$
C^*_{f,\theta_2}(A) \supseteq C^*_{f,\theta_1}(A)\,.
$$

Proof If we choose

$$
\tilde{f}(\alpha) = \frac{\theta_1(\alpha)f(\alpha)}{\int_0^1 \theta_1(\alpha)f(\alpha)\,d\alpha} \qquad \tilde{g}(\alpha) = \frac{\theta_2(\alpha)f(\alpha)}{\int_0^1 \theta_2(\alpha)f(\alpha)\,d\alpha}
$$

we obtain

$$
\varepsilon_f(\theta_2) - \varepsilon_f(\theta_1) = \int_0^1 \alpha\tilde{f}(\alpha)\,d\alpha - \int_0^1 \alpha\tilde{g}(\alpha)\,d\alpha\,.
$$

It is trivial to see that as θ_1 and θ_2 verify (9) then even relation (8) is true. If in Lemma 1 we choose the increasing function $h(\alpha) = \alpha$ we obtain

$$
\varepsilon_f(\theta_2) - \varepsilon_f(\theta_1) \geq 0.
$$

Moreover, if we choose the decreasing function $h(\alpha) = spr(A_\alpha)$ from Lemma 1 we get

$$spr(C^*_{f,\theta_1}(A)) - spr(C^*_{f,\theta_2}(A)) = \int_0^1 spr(A_\alpha)\tilde{f}(\alpha)\,d\alpha - \int_0^1 spr(A_\alpha)\tilde{g}(\alpha)\,d\alpha \le 0\,.$$

Since $mid(C^*_{f,\theta_1}(A)) = mid(C^*_{f,\theta_2}(A))$ we have $C^*_{f,\theta_1}(A) \subseteq C^*_{f,\theta_2}(A)$.

Property (9) means that for $\alpha \ge \gamma$ we have $\frac{\theta_1(\alpha)}{\theta_1(\gamma)} \ge \frac{\theta_2(\alpha)}{\theta_2(\gamma)}$. Thus, if θ_1 increases faster than θ_2, the approximation interval will have a smaller spread.

Corollary 1 (i) *If θ is constant then $\varepsilon_f(\theta) = 0$ and $C^*_{f,\theta}(A) = M_f(A)$;*
(ii) *if θ is a decreasing function then $\varepsilon_f(\theta) \ge 0$ and $C^*_{f,\theta}(A) \supseteq M_f(A)$;*
(iii) *if θ is an increasing function then $\varepsilon_f(\theta) \le 0$ and $C^*_{f,\theta}(A) \subseteq M_f(A)$.*

Proof Assertion (i) follows by observing that if $\theta_1 = \bar{\theta}$ is constant, with $\bar{\theta} \in [0, 1]$, then $\varepsilon_f(\bar{\theta}) = 0$ and $C^*_{f,\bar{\theta}}(A) = M_f(A)$. To prove (ii), assume that θ is decreasing. The functions $\theta_1 = \bar{\theta}$ (constant) and $\theta_2 = \theta$ satisfy the property (8). Thus, from Proposition 2 we have

$$\varepsilon_f(\theta) \ge \varepsilon_f(\bar{\theta}) = 0$$

and

$$C^*_{f,\theta}(A) \supseteq C^*_{f,\bar{\theta}}(A) = M_f(A)\,.$$

Assertion (iii) follows in a similar way, assuming θ increasing and applying Proposition 2 to the functions $\theta_1 = \theta$ and $\theta_2 = \bar{\theta}$.

6 Trapezoidal Fuzzy Numbers

In this section we are interested in showing some results for a nonmonotonic θ to have a more evident connection between $\varepsilon_f(\theta)$ and the approximation interval size. Starting with the particular case of a trapezoidal fuzzy number to reach our aims we present a particular parametric representation of $\theta(\alpha)$ that includes the increasing, decreasing, and nonmonotonic case.

For a trapezoidal fuzzy number A the α-cuts are

$$A_\alpha = [a_1 + \alpha(a_2 - a_1), a_4 - \alpha(a_4 - a_3))] \qquad 0 \le \alpha \le 1\,.$$

Observing that

$$mid(A_\alpha) = \frac{a_1 + a_4}{2} + \frac{a_2 - a_1 + a_3 - a_4}{2}\alpha$$

$$spr(A_\alpha) = \frac{a_4 - a_1}{2} - \frac{a_4 - a_3 + a_2 - a_1}{2}\alpha$$

we obtain from (2)

$$mid(M_f(A)) = \frac{a_1 + a_4}{2} + \frac{a_2 - a_1 + a_3 - a_4}{2} e_f$$
$$spr(M_f(A)) = \frac{a_4 - a_1}{2} - \frac{a_4 - a_3 + a_2 - a_1}{2} e_f$$

where e_f is defined in (5). Note that the f-weighted interval-valued possibilistic mean of a trapezoidal fuzzy number is the α-cut at the level of the preference index value e_f, that is $M_f(A) = A_{e_f} = [a_L(e_f), a_R(e_f)]$. The approximation interval $C^*_{f,\theta}(A)$ is given by

$$mid(C^*_{f,\theta}(A)) = mid(M_f(A)) \tag{10}$$

and, from (3),

$$spr(C^*_{f,\theta}(A)) = \frac{a_4 - a_1}{2} - \frac{a_4 - a_3 + a_2 - a_1}{2} k_{f,\theta}$$

where $k_{f,\theta}$ is defined in (6). Observe that $mid(C^*_{f,\theta}(A)) = mid(A_{e_f})$ and $spr(C^*_{f,\theta}(A)) = spr(A_{k_{f,\theta}})$. Furthermore, the larger $k_{f,\theta}$ is, the smaller the spread of approximation will be. We have from (7)

$$spr(C^*_{f,\theta}(A)) - spr(M_f(A)) = \frac{a_4 - a_3 + a_2 - a_1}{2} \varepsilon_f(\theta)\,. \tag{11}$$

From (10) and (11) we obtain the following general result for trapezoidal fuzzy numbers

Proposition 3 (i) *If* $\varepsilon_f(\theta) = 0$ *then* $C^*_{f,\theta}(A) = M_f(A)$*;*
(ii) *if* $\varepsilon_f(\theta) > 0$ *then* $C^*_{f,\theta}(A) \supset M_f(A)$*;*
(iii) *if* $\varepsilon_f(\theta) < 0$ *then* $C^*_{f,\theta}(A) \subset M_f(A)$.

6.1 Example

Let us consider for $0 < \beta < 1, n > 0$

$$\theta(\alpha) = \begin{cases} \left(\frac{\beta-\alpha}{\beta}\right)^n & \alpha < \beta \\ \left(\frac{\alpha-\beta}{1-\beta}\right)^n & \alpha \geq \beta \end{cases} \qquad \alpha \in [0, 1]$$

Note that if $\beta = 1/2$ we have $\theta(\alpha) = |1 - 2\alpha|^n$.
Furthermore, we may consider the limit cases $\beta = 0$ corresponding to the increasing

function $\theta(\alpha) = \alpha^n$, and $\beta = 1$ corresponding to the decreasing function $\theta(\alpha) = (1-\alpha)^n$. In the following we will denote $\varepsilon_f(n, \beta) = \varepsilon_f(\theta)$.

6.2 Case $f(\alpha) = 1$

In the case $f(\alpha) = 1$ we have $M_f(A) = EI(A)$ where $EI(A)$ is the expected interval. We obtain

$$e_f = \frac{\int_0^1 \alpha\, d\alpha}{\int_0^1 d\alpha} = \frac{1}{2}, \qquad k_{f,\theta} = \frac{\int_0^1 \alpha\, \theta(\alpha)\, d\alpha}{\int_0^1 \theta(\alpha)\, d\alpha} = \frac{(1-\beta)n+1}{n+2}.$$

If $C^*(A) = C^*(A; f, n, \beta)$ is the approximation interval, when $n \to 0$ we have $C^*(A) \to EI(A)$. From Proposition 3 we obtain by computation

(i) $\beta = 1/2 \implies \varepsilon_f(n, \beta) = 0$ (for all $n > 0$) $\implies C^*(A) = EI(A)$,
(ii) $\beta > 1/2 \implies \varepsilon_f(n, \beta) > 0$ (for all $n > 0$) $\implies C^*(A) \supset EI(A)$,
(iii) $\beta < 1/2 \implies \varepsilon_f(n, \beta) < 0$ (for all $n > 0$) $\implies C^*(A) \subset EI(A)$.

Numerical example. To show how the interval approximation proposed works, we consider the symmetric trapezoidal fuzzy number $A = (1, 2, 4, 5)$ shown in Fig. 1 and compute the approximation interval of A for different values of parameters β and n when $f(\alpha) = 1$. By computation we obtain $EI(A) = [\frac{3}{2}, \frac{9}{2}]$, $C^*(A) = C^*(A; f, n, \beta) = [c_L^*, c_R^*] = \left[\frac{(2-\beta)n+3}{n+2}, \frac{(4+\beta)n+9}{n+2}\right]$ and that the smaller β is, the smaller $C^*(A)$ will be. Fig. 2a shows the approximation interval, in the case $n = 1$, for each level β. When $\beta = 0.5$ we have $C^*(A) = EI(A)$. In Fig. 2b we have represented the interval approximation if $n = 0$ (continuous line), $n = 1$ (dashed line), $n = 2$ (dotted line), $n = \infty$ (dashed–dotted line). Note that when $n = 0$ we have $C^*(A) = EI(A)$ and when $n \to +\infty$ we have $C^*(A) \to [2-\beta, 4+\beta]$. Furthermore, n can be interpreted as an intensification parameter because when $n \to +\infty$ we have $C^*(A) \searrow$ if $\beta < 1/2$ and $C^*(A) \nearrow$ if $\beta > 1/2$.

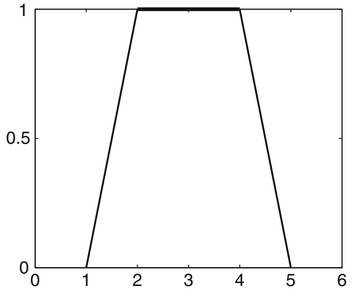


Fig. 1 Trapezoidal fuzzy number

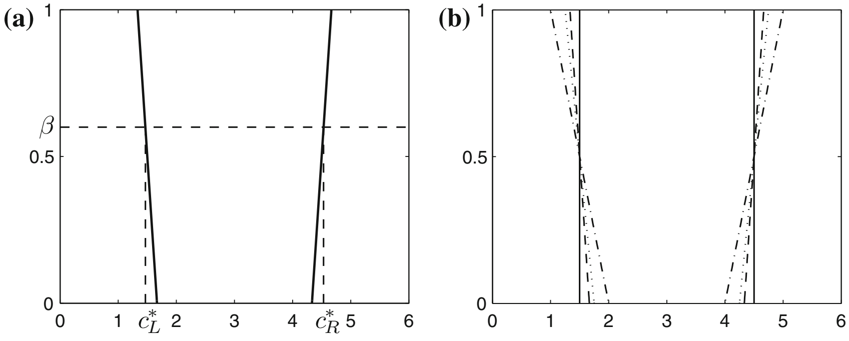


Fig. 2 Interval approximation

6.3 Case $f(\alpha) = \alpha$

In the case $f(\alpha) = \alpha$ we have $M_f(A) = M(A)$ where $M(A)$ is the interval-valued possibilistic mean. We get

$$e_f = \frac{\int_0^1 \alpha^2 \, d\alpha}{\int_0^1 \alpha \, d\alpha} = \frac{2}{3}$$

$$k_{f,\theta} = \frac{\int_0^1 \alpha^2 \, \theta(\alpha) \, d\alpha}{\int_0^1 \alpha \, \theta(\alpha) \, d\alpha} = \frac{(1-\beta)n^2 + (3-\beta-2\beta^2)n + 2}{(1-\beta)n^2 + (4-3\beta)n + 3} \, .$$

By computation we obtain that, for $n > 0$ fixed, the equation $\varepsilon_f(n, \beta) = 0$ has a unique solution in the interval (0, 1) given by

$$\beta_*(n) = \frac{1}{4} + \frac{\sqrt{n^2 + 18n + 33} - n}{12} \, .$$

The solution $\beta_*(n)$ is increasing with respect to n, $\lim_{n \to +\infty} \beta_*(n) = 1$ and

$$\lim_{n \to 0} \beta_*(n) = \frac{1}{4} + \frac{\sqrt{33}}{12} \, .$$

From Proposition 3 we obtain

(i) $\beta = \beta_*(n) \implies \varepsilon_f(n, \beta) = 0 \implies C^*(A) = M(A)$,
(ii) $\beta > \beta_*(n) \implies \varepsilon_f(n, \beta) > 0 \implies C^*(A) \supset M(A)$,
(iii) $\beta < \beta_*(n) \implies \varepsilon_f(n, \beta) < 0 \implies C^*(A) \subset M(A)$.

Remark 2 Note that for $\beta = 1/2$ in the case $f(\alpha) = 1$ we have $C^*(A) = EI(A)$ and in the case $f(\alpha) = \alpha$ we have $C^*(A) \subset M(A)$.

7 Conclusion

In this paper we have seen a new generalization of the Trutschnig et al. distance to evaluate the nearest interval to a fuzzy number. We conclude that the classical interval connected with a fuzzy number as $M_f(A)$ has the same middle point of the approximation interval we propose. So if we want to have the average value of a fuzzy number we may use the two methods indifferently. But if we want to have information on its ambiguity or other quantities connected with its spread, we obtain a different evaluation by working with the method we propose.
We are working in the direction to apply this result to the triangular and trapezoidal approximation of a fuzzy number.

References

1. Anzilli, L.: A possibilistic approach to investment decision making. Int. J. Uncertainty. Fuzziness Knowl. -Based Syst. **21**, 201–221 (2013)
2. Bede, B.: Mathematics of Fuzzy Sets and Fuzzy Logic. Studies in Fuzziness and Soft Computing, Vol. 295. Springer (2013)
3. Carlsson, C., Fuller, R.: On possibilistic mean value and variance of fuzzy numbers. Fuzzy Sets Syst. **122**, 315–326 (2001)
4. Chanas, S.: On the interval approximation of a fuzzy number. Fuzzy Sets Syst. **122**, 353–356 (2001)
5. Delgado, M., Vila, M.A., Voxman, W.: On a canonical representation of a fuzzy number. Fuzzy Sets Syst. **93**, 125–135 (1998)
6. Dubois, D., Prade, H.: The mean value of a fuzzy number. Fuzzy Sets Syst. **24**, 279–300 (1987)
7. Facchinetti, G.: Ranking functions induced by weighted average of fuzzy numbers. Fuzzy Optim. Decis. Mak. **1**, 313–327 (2002)
8. Fuller, R., Majlender, P.: On weighted possibilistic mean and variance of fuzzy numbers. Fuzzy Sets Syst. **136**, 363–374 (2003)
9. Grzegorzewski, P.: Nearest interval approximation of a fuzzy number. Fuzzy Sets Syst. **130**, 321–330 (2002)
10. Grzegorzewski, P.: On the interval approximation of fuzzy numbers. Advances in Computational Intelligence, 59–68. CCIS 299, Springer (2012)
11. Heilpern, S.: The expected value of a fuzzy number. Fuzzy Sets Syst. **47**, 81–86 (1992)
12. Liu, X.: On the maximum entropy parameterized interval approximation of fuzzy numbers. Fuzzy Sets Syst. **157**, 869–878 (2006)
13. Trutschnig, W., González-Rodrýguez, G., Colubi, A., Gil, M.A.: A new family of metrics for compact, convex (fuzzy) sets based on a generalized concept of mid and spread. Inf. Sci. **179**, 3964–3972 (2009)

Application of the Horizontal Membership Function to the Uncertain Displacement Calculation of a Composite Massless Rod Under a Tensile Load

Karina Tomaszewska and Andrzej Piegat

Abstract Fuzzy arithmetic, based on Zadeh's extension principle, is a common method applied to solve problems with uncertain parameters. The paper presents the fuzzy arithmetic operations on fuzzy numbers in a new way, using the horizontal membership functions (HMFs). The horizontal membership functions enable to introduce uncertain, interval, or fuzzy variable values together with crisp values in arithmetic operations without using Zadeh's extension principle. Thus, a relatively easy aggregation of crisp and uncertain knowledge is possible. The numerical example of one-dimensional static problem consisting of a two-component massless rod under tensile load is considered.

Keywords Fuzzy systems · Multidimensional RDM fuzzy arithmetic · Horizontal membership function · Uncertainty theory · Soft computing

1 Introduction

In solving real problems frequently uncertain data occurs. Calculating with it is difficult and requires special arithmetic. One of the modeling methods of uncertain data is the vertical membership function (vertical MF) introduced by Lotfi Zadeh and used in fuzzy arithmetic [1–3]. Example of a vertical MF is shown in Fig. 1.

The trapezium MF shown in Fig. 1 is expressed by (1):

$$\begin{aligned} x \in [a, b] &: \mu(x) = (x - a)/(b - a) \\ x \in [b, c] &: \mu(x) = 1 \\ x \in [c, d] &: \mu(x) = (d - x)/(d - c) \end{aligned} \tag{1}$$

K. Tomaszewska (✉) · A. Piegat
West Pomeranian University of Technology, Zolnierska 49, 71-210 Szczecin, Poland
e-mail: mtomaszewska@wi.zut.edu.pl

A. Piegat
e-mail: apiegat@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_6

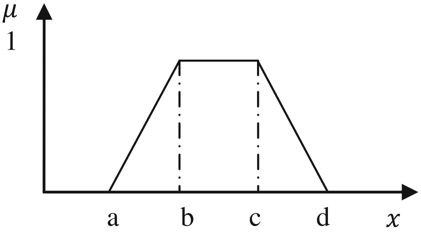


Fig. 1 A trapezium membership function (fuzzy interval)

It should be noted that the vertical MF is defined in 2D-space $X \times \mu$. The basic calculation method with MFs is Zadeh's extension principle [1–3]. If a problem is defined by the dependence $y = f(x_1, \ldots, x_n)$ and value of the dependant variable y for given values of independent variables $x_1, \ldots, x_n$ is to be calculated and part of values of variables x_i are known precisely and of other part are known only approximately in form of fuzzy interval A_i, then the resulting set $B(y)$ can be calculated with formula (2).

$$\mu_B(y) = \begin{cases} \sup_{z=f(x_1,\ldots,x_n)} \min\{\mu_{A_1}(x_1), \ldots, \mu_{A_n}(x_n)\} & if \quad \exists Z = f(x_1, \ldots, x_n) \\ 0 & otherwise \end{cases} \tag{2}$$

Calculation of MF $\mu_B(y)$ is complicated. It requires inspection of all n-tuples $\{x_1, \ldots, x_n\}$, calculation of membership for each of them, and determining the membership supremum. For, e.g., if function value $y = x_1/(x_2 + x_3)$ is to be calculated for uncertain variable values expressed by triangle MFs x_i (m,α,β), where m means the function typical (nominal) value, α means the left and β the right spread, then to determine the function $\mu(y)$ a great number of equalities of the type $y = 0.50 = x_1/(x_2+x_3)$, $y = 0.51 = x_1/(x_2+ x_3)$, $y = 0.52 = x_1/(x_2+ x_3)$, etc. All triples, which generate a given value y = const should be analyzed and then the triple with maximal membership $\mu(y =$ const), according to Zadeh's extension principle, should be determined. It requires a great calculation expenditure. Application of horizontal MFs allows for considerable decreasing of this expenditure. It also facilitates the use of MFs in all calculations. The author of the concept of horizontal MFs is Andrzej Piegat. This concept will be explained with an example of the trapezium MF, Fig. 2. However, horizontal MFs can be used to all types of MFs. Function $\mu(x)$ is unambiguous in the direction of variable μ, formula (1). And it is ambiguous in the direction of variable x.

Thus, in the classical understanding of functions it is not a function, because one value of μ is corresponded by two values of variable x: x_L (μ) and x_R (μ). Now, a new variable $\alpha_x \in [0; 1]$, is introduced. It has the meaning of a relative distance measure (RDM). This variable allows for determining of any point lying between two borders x_L (μ) and x_R (μ) of the function. On the left border variable $\alpha_x = 0$ and on the right border $\alpha_x = 1$. Between the borders α_x takes fractional values.

The idea of the RDM variable was used in the multidimensional RDM-interval arithmetic (RDM-IA) [4–7]. It allows for eliminating many paradoxes featuring

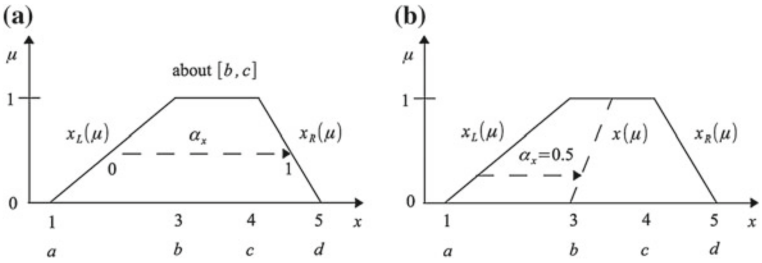


Fig. 2 Visualization of the horizontal approach to the mathematical description of the fuzzy interval about [*b*; *c*]

one-dimensional interval and fuzzy arithmetic. RDM-IA also shows that full and precise solutions of granular problems are multidimensional granules and in the general case they cannot be explained and understood on the basis of one-dimensional approaches. The left border x_L and the right border x_R of the membership function from Fig. 2 can separately and unambiguously be defined in the horizontal direction as $x_L(\mu)$ and $x_R(\ \mu)$, formula (3).

$$x_L = a + (b - a)\mu, \qquad x_R = d - (d - c)\mu \tag{3}$$

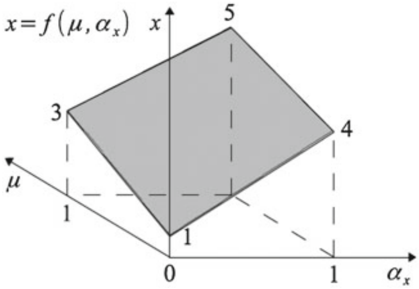
RDM variable α_x [0; 1], realizes a gradual transition of the left border x_L in the right border x_R. The transitional segment *x*(μ) in Fig. 2 is defined by function (4).

$$x = x_L + (x_R - x_L)\alpha_x;\ \alpha_x \in [0;1],$$

$$x = [a + (b - a)\mu] + [(d - a) - \mu(b - a + d - c)]\alpha_x \tag{4}$$

It should be noticed in formula (3) that $x\ = f(\ \mu, \alpha_x)$, which means that function (4) is a function of two variables and is defined in 3D-space. It can be seen in Fig. 3 that this function is unambiguous.

Fig. 3 The horizontal membership function $x\ = (1 + 2\ \mu) + (4 - 3\mu)\alpha_x$, $\alpha_x \in [0; 1]$, corresponding to the function from Fig. 2 in 3D-space as an unambiguous function



2 Numerical Problem

2.1 Description of the Problem

The problem concerns a composite massless rod under a tensile load. The rod is characterized by the length parameters $l^{(1)}$ and $l^{(2)}$, the cross sectional areas $A^{(1)}$ and $A^{(2)}$, and the Young's moduli $E^{(1)}$ and $E^{(1)}$ which quantify the elasticity of the rod's components. The rod is clamped at one end and is subjected to the tensile force F that acts as an external loading at the other end. To determine the displacement $u(x)$ of the cross-section at any position x within the rod, the well-established finite element method can be applied. In the paper, only one-dimensional structure with negligible body forces is considered; thus, Hooke's law is introduced:

$$\sigma = E\ \varepsilon \tag{5}$$

And the equation of virtual work for a single rod element i takes form:

$$E^{(i)} A^{(i)} \int_{l^{(i)}} \varepsilon^{(i)} \delta \varepsilon^{(i)} dx = F_j^{(i)} \delta u_j^{(i)} + F_k^{(i)} \delta u_k^{(i)} \tag{6}$$

The single finite element is shown in Fig. 4.

When we use linear shape functions for the displacement field $u^{(i)}(x)$ within the element i and we consider the strain-displacement relation:

$$\varepsilon^{(i)}(x) = \frac{d\ u^{(i)}(x)}{dx} \tag{7}$$

$$u^{(i)}(x) = \left[1 - \frac{x}{l^{(i)}}, \frac{x}{l^{(i)}}\right] \begin{bmatrix} u_j^{(i)} \\ u_k^{(i)} \end{bmatrix} = \boldsymbol{H}^{(i)} \boldsymbol{u}^{(i)} \tag{8}$$

$$\varepsilon^{(i)}(x) = \left[-1/l^{(i)}, 1/l^{(i)}\right] \begin{bmatrix} u_j^{(i)} \\ u_k^{(i)} \end{bmatrix} = \boldsymbol{D}^{(i)} \boldsymbol{u}^{(i)} \tag{9}$$

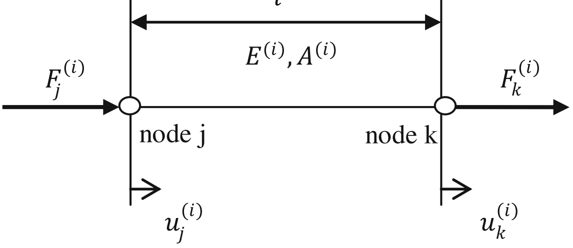


Fig. 4 The scheme of a rod's finite element

By introducing these relations into (5) and applying the fundamental lemma of variational principles, we obtain a linear system of equations in the matrix form:

$$\frac{E^{(i)}A^{(i)}}{l^{(i)}}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}=\begin{bmatrix}u_j^{(i)}\\ u_k^{(i)}\end{bmatrix}=\begin{bmatrix}F_j^{(i)}\\ F_k^{(i)}\end{bmatrix} \tag{10}$$

where we have the element stiffness matrix, the element displacement vector, and the result denotes the element nodal force vector. The assembly process for the system in Fig. 4 uses the condition of equilibrium of forces at node 2:

$$F_2^{(1)}+F_2^{(2)}=0 \tag{11}$$

As well as the continuity of displacements at that node, which requires $u_2^{(1)}=u_2^{(2)}$ and the boundary conditions $u_1=0$ and $F_3=F$.
It leads to the equation:

$$\begin{bmatrix}c^{(1)}+c^{(2)} & -c^{(2)}\\ -c^{(2)} & c^{(2)}\end{bmatrix}\begin{bmatrix}u_2\\ u_3\end{bmatrix}=\begin{bmatrix}0\\ F\end{bmatrix} \tag{12}$$

where

$$c^{(i)}=\frac{E^{(i)}A^{(i)}}{l^{(i)}}, i=1,2,\ldots$$

The existence of only two elements gives the possibility to solve the problem directly by eliminating either the displacement u_2 or the displacement u_3 and the procedure leads to:

$$\begin{aligned}u_2&=\frac{1}{c^{(1)}}F\\ u_3&=\left(\frac{1}{c^{(1)}}+\frac{1}{c^{(2)}}\right)F\end{aligned} \tag{13}$$

2.2 Solution for Fuzzy-Valued Parameters

We have to solve the finite element problem for a fuzzy-valued parameter configuration, where the first component of the rod is assumed to be steel with the material and geometry parameters [2]:

$$\begin{aligned}A^{(1)}&=100\,\text{mm}^{-2} & A^{(2)}&=75\,\text{mm}^{-2}\\ l^{(1)}&=500\,\text{mm} & l^{(2)}&=500\,\text{mm}\end{aligned}$$

The external load is specified by the force F=1000 N.

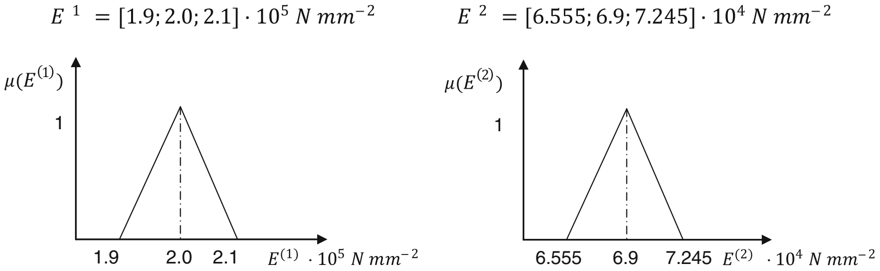


Fig. 5 The membership functions for the fuzzy numbers $E^{(1)}$ and $E^{(2)}$

The uncertain Young's moduli $E^{(1)}$ and $E^{(2)}$ are defined in the form:

$$\tilde{E}^{(i)} = gfn^*(\bar{E}_i, 0.05\bar{E}_i, 0.05\bar{E}_i), i = 1, 2.$$
$$\bar{E}^{(1)} = 2.0 \cdot 10^5 \,\mathrm{N\,mm^{-2}} \qquad \bar{E}^{(2)} = 6.9 \cdot 10^4 \,\mathrm{N\,mm^{-2}}$$

Making use of horizontal membership function, we can evaluate the fuzzy national expressions to determine the fuzzy-valued displacements. Let us consider the fuzzy numbers $E^{(1)}$ and $E^{(2)}$ with their membership functions shown in Fig. 5 [8].

According to formula (4) the fuzzy numbers $E^{(1)}$ and $E^{(2)}$ take form:

$$E^{(1)} = (1.9 + 0.1\mu + 0.2\alpha_1(1-\mu)) \cdot 10^5 N\ \mathrm{mm}^{-2}, \alpha_1 \in [0; 1] \tag{14}$$

$$E^{(2)} = (6.555 + 0.345\mu + 0.69\alpha_2(1-\mu)) \cdot 10^4 N\ \mathrm{mm}^{-2}, \alpha_2 \in [0; 1] \tag{15}$$

and using them in formula (13) we get:

$$u_2 = \frac{1}{38 + 2\mu + 4\alpha_1(1-\mu)} \tag{16}$$

$$u_3 = \frac{1}{38 + 2\mu + 4\alpha_1(1-\mu)} + \frac{2}{19.665 + 1.035\mu + 2.07\alpha_2(1-\mu)} \tag{17}$$

For the border cuts of μ the equation is different and takes formula:

$$\text{for } \mu = 0 \rightarrow u_2 = \frac{1}{38 + 4\alpha_1}$$

$$\text{for } \mu = 0 \rightarrow u_3 = \frac{1}{38 + 4\alpha_1} + \frac{2}{19.665 + 2.07\alpha_2} \tag{18}$$

For $\mu = 1$ the values of displacement are equal: $u_2 = 0.0250$ mm and $u_3 = 0.1216$ mm and they do not depend on values of α_1 and α_2.

Let us consider u_2 (Figs. 6 and 7):

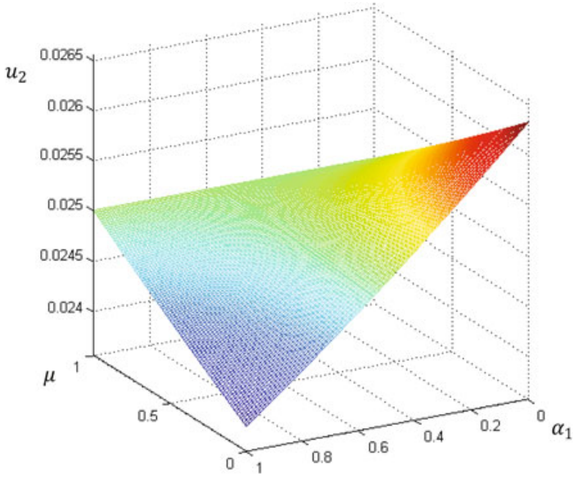


Fig. 6 Multidimensional result granule for u_2

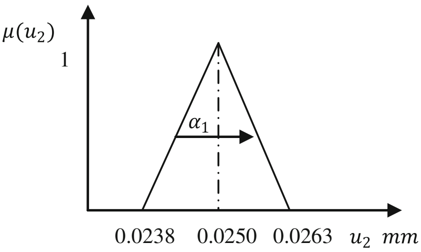


Fig. 7 The membership function for u_2

Let consider u_3:
Left border (for $\alpha_1 = 0$ and $\alpha_2 = 0$):

$$u_3 = \frac{1}{38 + 2\mu} + \frac{2}{19.665 + 1.035\mu} \tag{19}$$

Right border (for $\alpha_1 = 1$ and $\alpha_2 = 1$):

$$u_3 = \frac{1}{42 - 2\mu} + \frac{2}{21.735 - 1.035\mu} \tag{20}$$

If we rotate the shape which is based on the connection of left and right border, we get the vertical membership function for u_3, presented in Fig. 8.

Formula (17) shows that u_3 is not a one-dimensional function, it is $f(\mu, \alpha_1, \alpha_2)$ and defined in 4D-space. The distribution shown in Fig. 9 is not the result but only a 2D-information about occurrence possibility particular values of u_3 derived from the full 4D-result given by (17). On the basis of the exact solution also the representation delivered by Zadeh's extension principle can be achieved. The function $f(\mu, \alpha_1, \alpha_2)$ can be approximately shown in the form of a projection on 3D-space, where the

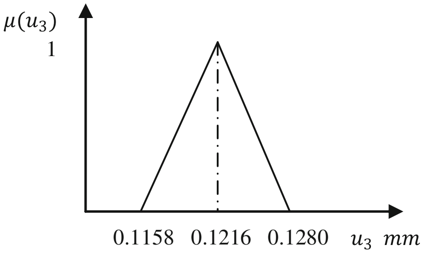


Fig. 8 The membership function for u_3 based on left and right border

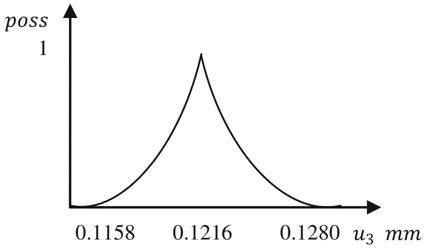


Fig. 9 Normalized possibility distribution of occurrence of particular values of u_3

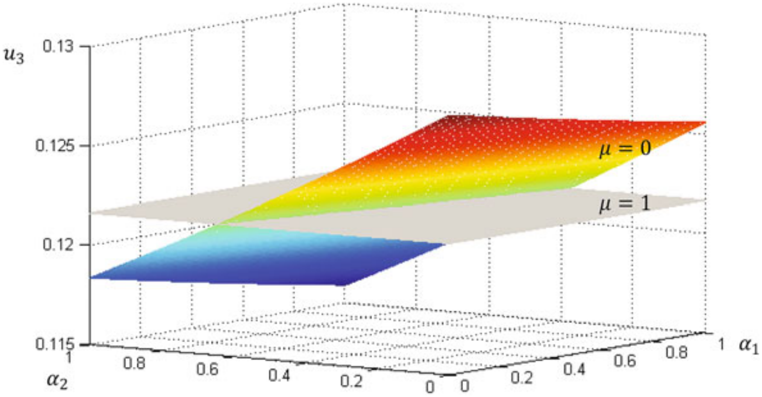


Fig. 10 Multidimensional result granule for u_3 being set of quadruples $\{\mu, \alpha_1, \alpha_2, u_3\}$ determined by formula (17) projected in 3D-space

dimension μ is shown in form of μ-cuts. In Fig. 10 a gray surface presents $u_3 = f(1, \alpha_1, \alpha_2)$ and a color surface is $u_3 = f(0, \alpha_1, \alpha_2)$.

Each of the contour lines is a set of infinite number of tuples satisfying the condition. As can be seen in Fig. 11, the value of $u_3 = 0.1280$ can occur only one time for $E_1 = 1.9$ and $E_2 = 6.555$. Instead, the value of $u_3 = 0.1216$ has considerably greater possibilities of occurrence because the number of tuples is infinitely large and two surfaces from Fig. 10 intersect at this line. The cardinality of particular sets

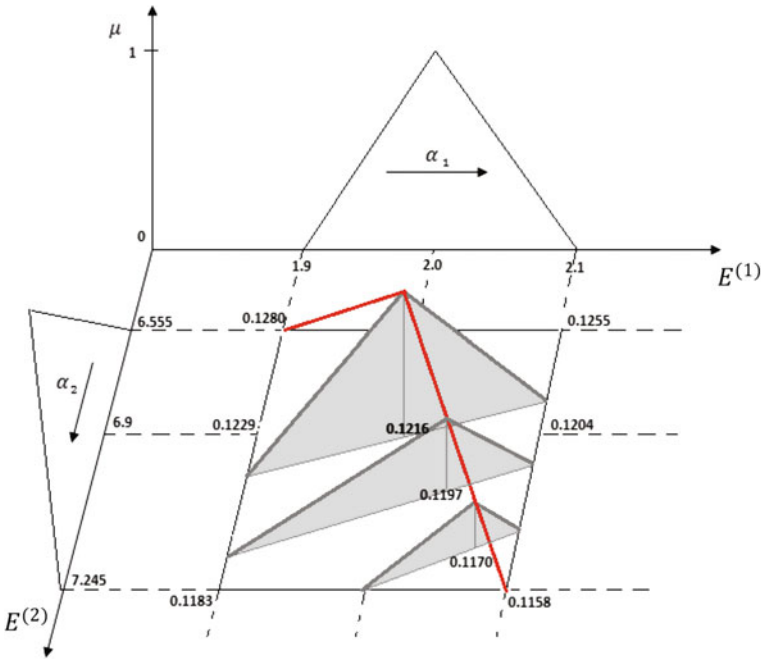


Fig. 11 The projection of the full 4D result granule of u_3 on 3D-space with slant contour lines

can be interpreted as absolute possibility of occurrence values (Fig. 9). The distribution from Fig. 8 corresponds in Fig. 10 to the edge going from the corner where $u_3 = 0.1280$ to the corner $u_3 = 0.1158$ through the peak of the membership function (a red curve).

3 Conclusions

This approach allows one to manage uncertainty by choosing a policy that delivers an acceptable performance for a known range of parameter outcomes. Horizontal membership functions can be a good method to make operations with many fuzzy numbers. HMFs can be inserted into classical mathematical formulas of the type $y = f(x_1, \ldots, x_n)$ together with crisp data items and achieved calculation results can be interpreted and understood. The main advantage of HMFs is that solving problems with their use does not require application of Zadeh's extension principle. Horizontal membership functions considerably facilitate fuzzy calculations and deliver multidimensional problems-solutions and not only representation of these solutions of lower dimensionality. The author believes that this paper is a good basement to lead the further researches showing an assessment of this method applicability.

References

1. Gupta, M., Kaufmann, A.: Introduction to Fuzzy Arithmetic. Van Nostrand Reinhold, New York (1991)
2. Hanss, M.: Applied Fuzzy Arithmetic. Springer, Berlin (2005)
3. Piegat, A.: Fuzzy Modeling and Control. Physica-Verlag, New York (2001)
4. Piegat, A., Landowski, M.: Is the conventional interval-arithmetic correct? J. Theor. Appl. Comput. Sci. **6**, 27–44 (2012)
5. Piegat A., Landowski M.: Horizontal membership function and examples of its application. Paper was sent in February to International Journal of Fuzzy Systems, under review
6. Piegat A., Plucinski M.: Computing with words with use of inverse RDM-models of membership functions. Paper was sent in February 2014 to International Journal of Applied Mathematics and Computer Science, under review
7. Piegat, A., Tomaszewska, K.: Decision-making under uncertainty using Info-Gap theory and new multidimensional RDM interval arithmetic. Electrotech. Rev. **89**(8), 71–76 (2013)
8. Zadeh, L.A: Fuzzy sets. Inf. Control. (1965)

Multi-agent Processes Analysis System in Prediction Task

Michał Zabłocki

Abstract In the article, author presents the concept of the system of process analysis based on the multiagent platform. The system implements a novel approach to the computer analysis complex processes. The distributed multiagent system studied in this paper is based on the decomposition of a complex task on a series of atomic tasks, as they are called, which are assigned to individual agents. The results of cooperation of those agents (of various specializations) within a single computer system allowing efficient solution of prediction of the process under consideration is also examined and discussed in this article.

Keywords Multiagent system · Processes analysis · PSO · Computer system decomposition · Prediction

1 Introduction

The analysis of processes of different kind is one of the central problems in computer simulation, as a whole, and data mining, in particular. Processes are customary studied in the context of a certain physical system, because physical essence of process components tells a good deal about the structure of the object analyzed and makes it possible to estimate functional dependences between the components. The latter allows the designer to build an efficient algorithm for computer simulation of the studied process. At the same time, we can formally consider the process as a dynamic system and apply mathematical and computer instrumentation to build efficient procedures for the analysis and prediction of future development of processes considering time series describing their states in discrete time instances.

M. Zabłocki (✉)
Faculty of Computer Science and Information Technology,
West Pomeranian University of Technology,
ul. Żołnierska 52, 71-210 Szczecin, Poland
e-mail: mzablocki@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_7

The task of process analysis is a complex issue, especially in the case when process components are strongly interrelated. To form a mathematical model of any process, we should carry out a series of observations over the process and be sure that the analyzed probes are sufficient to form the adequate model. It is possible if the process possesses certain stable statistic characteristics, say, it is stationary.

As is known, the mentioned issues make an important part of the theory of stochastic processes. But in the paper the author focuses on some computing aspects concerned with structural organization of the system dedicated to the analysis of complex processes by distributed system means, namely, systems based on the principles of multiagent systems.

To build mathematical model of the analyzed system, the regression function can be used. The statistical method based on building regressions allows us to study the relationships between data values. The task of constructing a regression model comes down to define the function, which describes a relationship between the values of dependent variable and the explanatory variable.

The system presented in this paper was implemented on the basis of multiagent platform JADE [1]. Multiagent programming paradigm makes it possible to decompose and distribute complex tasks. Decomposition (a process of dividing a large task on atomic subtasks) allows distribution of subtasks between agents. Agents can specialize in the field of their operations. The main advantages of agent-based programming, such as full autonomy of the agents, highly efficient communication protocol, and an ability to perceive the external environment [2] have significant impact on the implemented system. Therefore, the multiagent technology has wide application in a distributed processing and grid systems [3].

2 Related Works

In the paper [4] authors suggest to the employee the multiagent system based on the forecasting algorithm to take into account different characters of distinct segments of nonstationary financial time series. They present a method, which can help to reduce one step ahead forecasting error. They use a system of several adaptive forecasting agents instead of single prediction rule. In this system, agents evolve and compete among themselves. Final decision is made by a collective of the most successive agents.

The authors of [5] discuss the problem of predicting financial market movements. They were especially interested in the concept of artificial market model which has been trained on real financial data. They analyzed the theoretical machinery involved in the process of building such model. It resulted in a description of detailed methodology, which uses optimization techniques to estimate the parameters of the strategy distributed across the multitrader population. In this system, agents play the role of traders that estimates parameters of a predicting model using the Kalman Filter with constraints.

The publication [6] deals with the issue of multiagent forecasting in time series. It draws from literature on time series, graphical models, and multiagent systems. In a proposed system, knowledge representation of the agents is based on dynamic multiple-sectioned Bayesian networks (DMSBNs), a class of cooperative multiagent graphical models. Therefore, agents can perform one-step forecast with exact probabilistic inference.

3 Processes Analysis Task

A process identification is the main component of the task of process analysis. It uses a statistical methods in the purpose of building mathematical models of dynamical systems from measured data. The identification is an iterative set of actions presented in Fig. 1. In this figure, the stage associated with selection of the data lengths of learning and testing phase is novel in comparison with standard identification [7]. Machine learning methods are applied to compute optimal data length of learning and testing phases, model structure, and his parameters.

First stage of analysis is simple and comes down to reading data from file, in which are collected the results of measurements observed phenomenon. Second stage involves transformation of read data to a form that permits the processing of the system. Third stage is very important from the system point of view. In this stage are computed a optimal lengths of time series used in learning and testing phase. Next two stages are associated with learning phase. In the learning phase, time series, whose length were adjusted is used for the purpose of finding proper model structure and his parameters. The last stage is connected with testing phase, in which the efficiency of the model is tested. In learning phase, the optimization task comes down to finding the best time series length, for which the best model structure and his parameters will be chosen in order to best fit model to analyzed process (see Fig. 2). Whereas in the testing phase, the model will be applied according to its purpose for computing in this stage the period of time (see Fig. 3).

In the fourth stage on the optimal length of learning time series is chosen the structure of the model, which in the best way will give the nature of analyzed process.

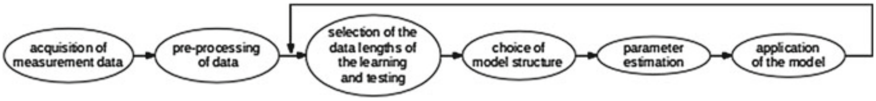

acquisition of measurement data
pre-processing of data
selection of the data lengths of the learning and testing
choice of model structure
parameter estimation
application of the model

Fig. 1 The schema of process analysis stages

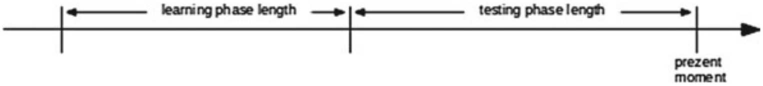


Fig. 2 Choice of data length of learning and testing phases

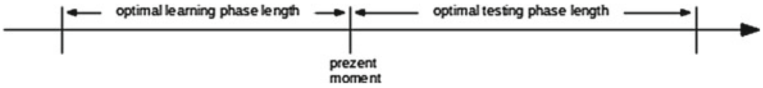


Fig. 3 Learning and testing phases

In the fifth stage for chosen model structure are computed best parameters. In the sixth stage, the model is applied on earlier computed period of time. At this point, the iteration loops back, and again the first step is performed.

Processes analysis is performed for couple different purposes. The analysis may in result gives control model, prognostic model, or simply a model for understanding the observed phenomenon. For the purpose of realization, in this publication task the predictive models are built and are used to predict future course of process.

4 Choice of Model Structure and Parameters

The system is mainly based on machine learning. The main goal of machine learning is a practical application of artificial intelligence methods for the purpose of creating the system that is able to improve himself with the help of the collected data. Thanks to appropriate algorithms, the intelligent system should be able to collect a knowledge and reasoning on the basis of this knowledge [8]. The data mining techniques are strongly associated with the concept of machine learning. Its permits to explore huge amount of data in the purpose of discovering patterns and regularity in such data. Very often data mining methods based on machine learning methods leads to building a model, which is described by analyzed dataset.

Described in this publication system uses linear regression model. Linear regression model is based on polynomial function. This function in many cases, in relatively simple way, very well describes dependencies between data. In this case, the model will take the form:

$$y = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_k x_k + \varepsilon\,, \tag{1}$$

where y is called dependent variable, x_i are called explanatory variables, a_i are called regression coefficients, and ε is called the error term.

To determine the appropriate function for this type of regression the method of least squares is used—mainly because of the simplicity of calculation. Regression model is constructed to best fit to the data collected during the observation of the analyzed process. This data takes the form of time series where in constant time step were measured the value of analyzed physical phenomenon. This is the so-called learning data and is used in machine learning phase in the model building process.

In the case of implemented system, the main goal of machine learning phase is to match an appropriate model structure and finds the best values of model parameters.

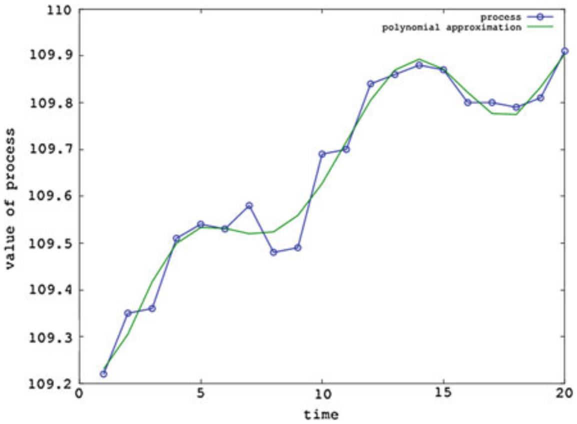


Fig. 4 Subsequent values of process and its approximation by polynomial function of eighth degree

This is typical optimization task. To find the solution of this task, the particle swarm optimization (PSO) algorithm were applied. Authorship of the algorithm originally is attributed to Kennedy, Eberhart, and Shi [9, 10]. But the idea of swarm intelligence were set out in the book by Kennedy and Eberhart [11]. This is a computational optimization method, which in iterative manner attempts to improve the result of goal function according to quality criteria assessment. PSO solves the problem using a population of potential solutions, called particles. It moves them in the space limited by the ranges of the variables of the objective function. The new position of a particle depends on its previous position and velocity, while taking into account the local and global best position in the space of possible solutions.

The objective function assumes to obtain the greatest amount of accurate forecasts for a given time interval. So the degrees of the polynomial are matched. Then the length of the time series is adjusted, which will serve to estimate best model parameters. Figure 4 presents subsequent values of process and its approximation by polynomial function of eighth degree.

5 Application of the Model

The processes analysis system builds a predictive model in order to forecast the future course of the analyzed process. At this point, begs the question of how long will the model and its parameters properly fulfill their task. The answer for this question is not so obvious because it depends on the nature of the analyzed process. There are two types of processes: stationary process and dynamic process (nonstationary). The stationary process is a stochastic process, in which joint probability distribution is constant during the time shift. Thus, its mean value and variance (if they are present), also are constant over time and do not follow any trends. Usually processes are not strictly stationary. The processes that occur in nature are more or less dynamic.

Therefore, the mean value and variance are changing and the model of such process will fulfill its purpose only in short period of time. During the stage of choosing the data length of testing phase, the implemented system examines the nature of the process and try to fit optimal usability length of the model. It assumes that strongly stationary processes have longer period of usability of model and nonstationary processes have very short period of usability of model.

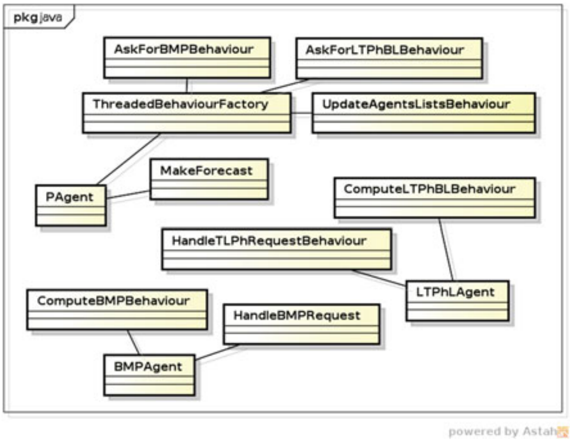
Author of the system has taken the assumption of nonstationarity of analyzed processes. According to this assumption, the author has awareness that implemented solutions theoretically do not guarantee that models, but when computed during the analysis, will be able to meet its task. However, system assumes that the model built based on historical data, is able to fulfill its task in future (in some period of time). Reference [12] As it is shown in an experiment, the implemented system works and gives good results (see Fig. 8).

6 Multiagent Decomposition of Processes Analysis Task

Implementation of the system is based on a multiagent platform. It means that the task of process analysis is divided into subtasks that correspond to the specific stages of analysis. In the system, we can distinguish three types of agents: an agent, which calculates optimal data length for learning and testing phases, hereinafter referred as LTPhLAgent; an agent that matches optimal structure of the model and its parameters, called BMPAgent; and an agent that performs forecasts, called PAgent. Each agent has a set of behaviors that allow it to efficiently communicate with other agents (see Fig. 6) and realize entrusted them tasks (see Fig. 5).

LTPhLAgent has two behaviors. The first is associated with the calculation of the optimal data length of learning and testing phases. The second behavior is associated with receiving information from other agents, and sending back responses

Fig. 5 Agents class diagram



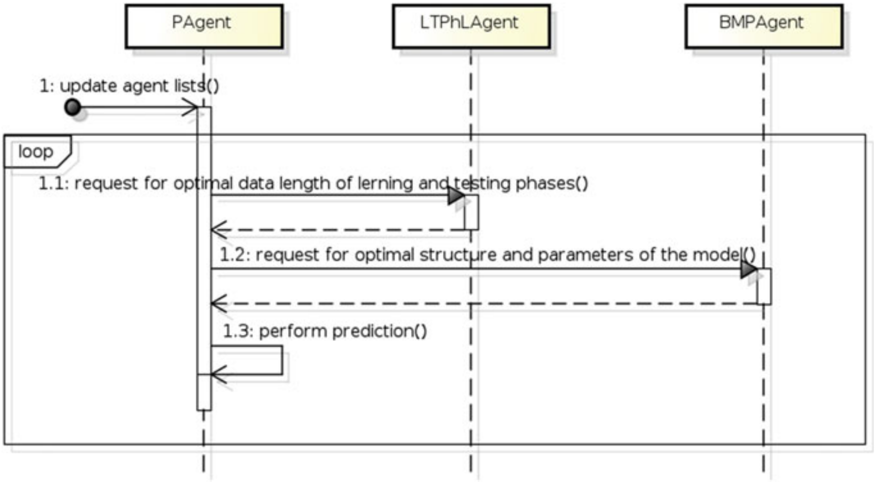

PAgent
LTPhLAgent
BMPAgent
1: update agent lists()
loop
1.1: request for optimal data length of lerning and testing phases()
1.2: request for optimal structure and parameters of the model()
1.3: perform prediction()

Fig. 6 Diagram of agent interactions

resulting from the calculation. When the agent receives the information with a request to make deductions, updates the parameters of calculation algorithm and performs calculations.

BMPAgent and LTPhLAgent have similar tasks. The behavior mechanism of BMPAgenta is analogous to the above-described. The only difference lies in the fact that BMPAgent computes optimal structure and parameters of the model.

The most tasks are assigned to the PAgent. Simultaneously, it implements the most different behaviors. This agent sends a request for calculation of the optimal data length of learning and testing phases to the LTPhLAgent. Afterward it receives and interprets the answer for this request. On the basis of received information, it sends a request for calculation of the optimal structure and parameters of the model to the BMPAgent. Then again it receives and interprets the second answer. Finally, it performs prediction and interprets its results.

7 The Role of the Ontology

For the purposes of describing the multi-agent system in this paper, a formal representation of knowledge obtained during the analysis of a process has been developed. This required the creation of a set of concepts and relationships between them. Notation of this collection creates a conceptual framework, which provides a description of the resulting knowledge. It is used for inference and supports processing of information carried by the ontology. It is the basis for effective communication between agents operating within the system. The developed ontology provides the meaning to the information obtained from the analysis of the process.

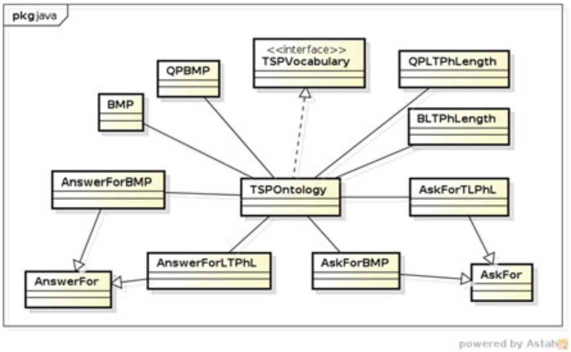


Fig. 7 The structure of the ontology

It should be emphasized that this is an example of domain ontology driven by system requirements, in this case related to the field of knowledge associated with being built model of the system of analyzed process.

In particular, the ontology of this system fulfills the following roles: defines an object that describes the parameters required to calculate the optimal data length of learning and testing phase used by the LTPhLAgent; defines an expression that describes the request PAgent addressed to LTPhLAgenta; defines an object that describes the result of calculations made by LTPhLAgent; defines an expression that describes the answer of the LTPhLAgent; defines an object that describes the parameters for BMPAgent, required to calculate the optimal structure and parameters of the model; defines an expression that describes the request PAgent addressed to BMPAgenta; defines an object that describes the result of calculations made by BMPAgent; and defines an expression that describes the answer of the BMPAgent (see Fig. 7).

8 Experiment

The main experiment was made using implemented multiagent system for processes analysis. The main goal was to check the operation of the system. In the experiment time series were used which constitute an hourly course of the closing price of a currency pair EUR/JPY in range from August 23, 2011 from 18:00 to April 1, 2013, 01:00. The data has a length of 9895 h and are stored in a file named EURJPY60.csv.

After the start of the system, it launches a multiagent platform and places on it a user-defined number of agents. The system for proper operation needs at least three agents: PAgent, BMPAgent, and LTPhLAgent. Each agent must register in yellow pages directory. In addition, BMPAgent and LTPhLAgent must register their services. PAgent searches for agents (their services) in the directory and groups them in terms of the services they offer. Then sends a request for the optimal data length of learning and testing phase to the LTPhLAgents. Having gained at least one answer, sends a request for the optimal structure of the model and its parameters to BMPAgents.

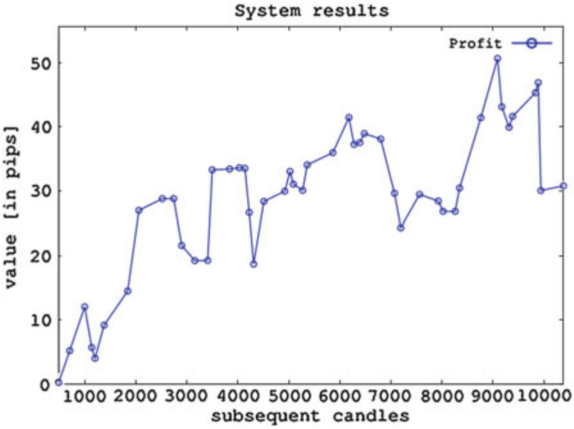


Fig. 8 Cumulative profit after using the system in a simple investment strategy

Finally on the basis of responses received from them makes a prediction of the future course of the analyzed process. The results of prediction is compared with the actual course of the process and restarts the process of building the model for another period.

The results of the analysis of input data were applied in a simple investment strategy. The strategy was based on information obtained from the system. If the system predicted the rise in the price of exchange rate of analyzed currency pair, it was opened so-called long position.[1] Otherwise, the so-called short position was opened. It should be emphasized that, in order to simulate the real conditions of the trading game, from return of the transaction was deducted the cost of the transaction which is normally deducted by the broker. The results of applied strategy are introduced in Fig. 8.

9 Additional Studies

The described multiagent processes analysis system has been tested for efficacy in selecting the data length of learning and testing phases. It has been checked how the lengths of learning and testing phases have been changing over the entire dataset. It is presented in Table 1. Additionally, the rows of table where the system correctly matches the length, i.e., it was returned profit, has been marked. By counting the number of correctly matched lengths and juxtaposing them with the amount of all periods, we can determine the average overall efficiency of the system at the level of 58.69 %. It should also be noted that both lengths of data (in learning and testing phase) are constantly changing. This proves that the parameters of the analyzed process are continuously changing, and so confirmed the validity of the undertaken calculations.

[1] *long position*—in the financial market is the purchase of a financial instrument or possession of the financial instrument on relevant account. *short position*—in the financial market means the sale of a financial instrument.

Table 1 Data length of learning and testing phases in subsequent candles

Candle number	Learning Phase Length	Testing Phase Length	Candle number	Learning Phase Length	Testing Phase Length	Candle number	Learning Phase Length	Testing Phase Length
489	637	217	4139	962	86	7195	664	369
706	782	293	4225	890	86	7564	454	366
999	839	138	4311	483	200	7930	872	96
1137	816	64	4511	795	410	8026	582	234
1201	809	178	4921	763	103	8260	690	88
1379	526	462	5024	866	65	8348	726	417
1841	668	220	5089	881	183	8765	530	327
2061	583	460	5272	618	90	9092	612	83
2521	679	223	5362	528	500	9175	830	143
2744	942	157	5862	611	317	9318	765	71
2901	969	257	6179	703	100	9389	484	447
3158	632	251	6279	974	113	9836	656	54
3409	959	91	6392	715	88	9890	621	51
3500	737	339	6480	992	323	9941	832	443
3839	998	191	6803	814	269	10384	829	159
4030	972	109	7072	770	123			

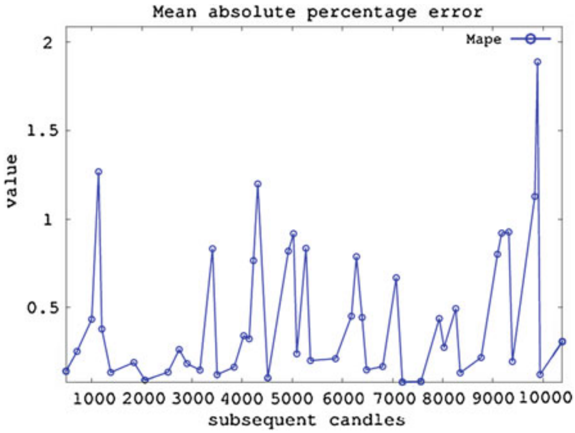
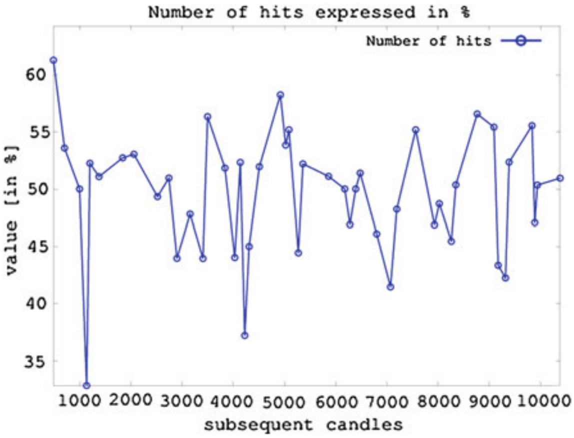


Fig. 9 Mean absolute percentage error

Then the Mean Absolute Percentage Error (MAPE) indicators [13] were analyzed. Figure 9 shows the change in the value of this indicator over the testing periods. The average value of this indicator throughout the range of data is equal approximately 0.45.

At the end, the numbers of hits in each period of testing phase was analyzed. Figure 10 presents the number of hits in a given testing period, expressed in percentage (divided by the length of the period). The average value in this case is equal 49.59 %. This value points at the varying weak forecast effectiveness of the

Fig. 10 Hits number expressed in percentage



system. However, this graph needs to be considered in a slightly different manner. It is necessary to remember that each testing period was preceded by new calculations. These calculations in any manner were not associated with the preceding calculations. Therefore, each period should be considered separately. If we count all periods where accuracy exceeded 50 % (there are 25) and divide them by the total number of testing periods, we get a value of 0.5438 (54.38 %). In conclusion, the efficiency of the system exceeds 54 %, which is a good result. It is confirmed by the profit graph of the conducted experience (Fig. 8).

10 Conclusions and Future Works

The decomposition of process analysis task into subtasks in order to perform them independently constitutes the foundation for the multiagent processes analysis system. Thereby, it was possible to assign these subtasks to individual agents. By using an agent-oriented programming paradigm implemented system acquired unique features that improves the processes analysis calculations. Multiagent technology made the system intelligent. Agents, through the developed ontology, became aware of their operations. They not only were able to immunize the system on calculation errors, but mainly improved the calculations by consciously monitoring them.

The PSO algorithm has a great impact on an optimization task. This highly effective method of searching for extrema of function significantly accelerated the calculations. However, the downside of this method is the large number of parameters. Appropriate adjustment of them to the needs of the optimization task is quite a complicated task. This method seeks the best solution by random wandering. Inappropriate tuning of this tool in extreme cases may result in fail. Finally, it turned out that the objective function is also important element, where besides maximizing the number of hits were also possible other goals such as maximizing the cumulative profit or minimizing the MAPE. Each of these criteria in various manner (positive or negative) may affect the results.

The experiment shows that the system based on the methods of data mining and machine learning is able to meet the challenge that goes beyond human ability. Its so-called computational intelligence allows to create a tool that is able to solve problems exceeding the capabilities of natural human intelligence. As shown in the conducted analysis, in spite of the fact that the system makes mistakes in individual forecasts, the resultant of system performance is positive.

The implemented system is one of the many proposals for decomposition of processes analysis task. Future studies will be aimed primarily at reducing the impact of nonstationarity of the process on the model. This will lead to adding a mechanism of continuous adaptation of the model instead of building it from scratch.

On the other hand, the organizational structures of a multiagent system will be studied in order to investigate the impact of different types of organizations on the performance of the entire system in scope of process analysis tasks.

References

1. Bellifemine, F., Caire, G., Grenwood, D.: Developing Multi-Agent Systems with JADE, p. 286. Wiley (2007). ISBN-13: 978-0-470-05747-6
2. Wooldridge, M.: An Introduction to Multiagent Systems, pp. 348. Wiley, Chichester (2002). ISBN 0-471-49691 X
3. Rogoza, V., Zabłocki, M.: Grid computing and Cloud computing in Scope of JADE and OWL Based Semantic Agents—A Survey, Przegląd Elektrotechniczny, vol. 90, no. 2 (2014). ISSN 0033–2097
4. Raudys, S., Zliobaite, I.: The Multi-Agent System for Prediction of Financial Time Series—ICAISC 2006. LNAI 4029, pp. 653–662. Springer (2006)
5. Gupta, N., Hauser, R., Johnson, N.F.: Inferring the Composition of a Trader Population in a Financial Market. Econophysics of Markets and Business Networks, New Economic Windows. Springer, Milan (2007)
6. Xiang, Y., Smith, J., Kroes, J.: Multiagent bayesian forecasting of structural time-invariant dynamic systems with graphical models. Int. J. Approx. Reason. **52**(7), 960–977 (2011)
7. Isermann, R., Münchhof, M.: Identification of Dynamic Systems: An Introduction with Applications, vol. XXV, p. 705. Springer (2011). ISBN: 978-3-540-78878-2
8. Murphy, K.P.: Machine Learning: A Probabilistic Perspective. Mass, Cambridge (2012)
9. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks IV, pp. 1942–1948 (1995). doi:10.1109/ICNN.1995.488968
10. Shi, Y., Eberhart, R.C.: A modified particle swarm optimizer. In: Proceedings of IEEE International Conference on Evolutionary Computation, pp. 69–73 (1998)
11. Kennedy, J., Eberhart, R.C.: Swarm Intelligence, Morgan Kaufmann (2001). ISBN 1-55860-595-9
12. Tsay, R.S.: Analysis of Financial Time Series, Wiley (2005). ISBN 0-471-690740
13. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. Wiley, New York (2001)

Method of Static Classifiers Selection Using the Weights of Base Classifiers

Robert Burduk

Abstract The choice of a pertinent objective function is one of the most crucial elements in static ensemble selection. In this study, a new approach of calculating the weight of base classifiers is developed. The values of these weights are the basis for the selection process of classifiers from the initial pool. The obtained weights are interpreted in the context of the interval logic. A number of experiments have been carried out on several datasets available in the UCI repository. The performed experiments compare the proposed algorithms with base classifiers, oracle, sum, product, and mean methods.

Keywords Classifier fusion · Interval logic · Static classifiers selection · Multiple classifier system

1 Introduction

The pattern recognition task is one of the trends of research on machine learning [1]. In the case of the supervised classification, we have a set of data in which a class label is assigned for each observation. In this issue we can consider a lot of research trends that are associated with problems such as: feature selection, extraction of features, selection of the training set, classifier selection, and more. The classification task can be accomplished by a single classifier or by a team of classifiers. In the literature, the use of multiple classifiers for a decision problem is known as multiple classifier systems (MCS) or an ensemble of classifiers EoC [3, 8, 24]. These methods are popular for their ability to fuse together multiple classification outputs for better accuracy of classification.

The output of an individual classifier can be divided into three types [16].

- The abstract level—the classifier ψ assigns the unique label j to a given input x.
- The rank level—in this case for each input x, each classifier produces an integer rank array. Each element within this array corresponds to one of the defined

R. Burduk (✉)
Department of Systems and Computer Networks, Wroclaw University of Technology,
Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland
e-mail: robert.burduk@pwr.wroc.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_8

class labels. The array is usually sorted and the label at the top being the first choice.

- The measurement level—the output of a classifier is represented by a measurement value that addresses the degree of assigning the class label to the given output x. An example of such a representation of the output is a posteriori probability returned by Bayes classifier.

According to these three types of outputs of the base classifier, various problems of combination function of classifiers outputs are considered. The problems studied in [17, 22] belong to the abstract level. The combining outputs for rank level are presented in [11] and problems studied in [14, 15] belong to the last level.

The selection of classifiers is one of the important problems in the creation of EoC [12, 21]. This task is related to the choice of a set of classifiers of all the available pool of classifiers. Here you can distinguish static or dynamic selection [18, 23]. In static classifier selection one set of a classifier is selected to create an EoC. This EoC is used in the classification of all the objects from the testing set. The main problem in this case is to find a pertinent objective function for selecting the classifiers. One of the best objective functions for the abstract level of classifier outputs is the simple majority voting error [20]. In the dynamic classifier selection, for each unknown sample a specific subset of classifiers is selected [2]. It means that we are selecting different EoCs for different object from the testing set. In this type of the classifier selection, the classifier is chosen and assigned to the sample based on different features [25] or different decision regions [4, 13].

In this work we will consider a static approach to build the EoC. In detail, we propose the new method to select the classifiers from the available pool. This method is based on the correction of base classifiers and can be interpreted in the contents of the interval logic. The presented results are compared with the oracle concept [5] and base classifiers. The oracle classifier is used as the possible upper limit of classification accuracy of the EoC. As a fusion function for classifier outputs we use the sum and weighted sum methods.

The text is organized as follows: in Sect. 2 the ensemble of classifiers and combination functions of classifiers outputs are presented. Section 3 contains the new method for assigning weights of individual base classifiers and the proposed static selection of classifiers. Section 4 includes the description of research experiments comparing the suggested algorithms with base classifiers, oracle, sum, product, and mean methods. Finally, conclusions from the experiments are presented.

2 Ensemble of Classifiers

Let us assume that we possess K of different classifiers $\Psi_1, \Psi_2, \ldots, \Psi_K$. Such a set of classifiers, which is constructed on the basis of the same learning sample is called an ensemble of classifiers or a combining classifier. However, any of Ψ_i classifiers is described as a component or base classifier. As a rule K is assumed to be an odd number and each of Ψ_i classifiers makes an independent decision. As a result, of all the classifiers' action, their K responses are obtained. Having at the disposal a set of base classifiers one should determine the procedure of making the ultimate decision regarding the allocation of the object to the given class. It implies that the output information from all K component classifiers is applied to make the ultimate decision.

Combination function of classifiers outputs. In this work we consider the situation when each base classifier returns the estimation of a posteriori probability. This means that output of all the base classifier is at the measurement level. Let us denote a posteriori probability estimation by $\hat{p}_k(i|x)$, $k = 1, 2, \ldots, K$, $i = 1, 2, \ldots, M$, where M is the number of the class labels. One of the possible methods for such outputs is a linear combination method. This method makes use of linear function like Sum, Prod, or Mean for the combination of the outputs. In the sum method, the score of the group of classifiers is based on the application of the following sums:

$$s_i(x) = \sum_{k=1}^{K} \hat{p}_k(i|x), \qquad i = 1, 2, \ldots, M. \tag{1}$$

The final decision of the group of classifiers is made following the maximum rule and is presented accordingly, depending on the sum method (1):

$$\Psi_S(x) = \arg\max_i s_i(x). \tag{2}$$

Similarly, in the mean method we use the following mean:

$$m_i(x) = \frac{1}{K} \sum_{k=1}^{K} \hat{p}_k(i|x), \qquad i = 1, 2, \ldots, M, \tag{3}$$

and in the product method the following products are used:

$$p_i(x) = \prod_{k=1}^{K} \hat{p}_k(i|x), \qquad i = 1, 2, \ldots, M. \tag{4}$$

Now the final decision of the ensemble of classifiers is made according to the mean rule:

$$\Psi_M(x) = \arg\max_i m_i(x), \tag{5}$$

or the product rule:

$$\Psi_P(x) = \arg\max_i p_i(x). \tag{6}$$

In the presented methods (2), (5), and (6) discrimination functions obtained from the individual classifiers take an equal part in building the combined classifier. Also, the weighted versions of these methods can be created. In this approach, each of the classifiers have an allocated weight, which is taken into account by reaching the final decision of the group. Weights depend largely on the quality of their base classifiers. In the case when each classifier has one weight for all the possible classes, an adequate group classification formula for the sum method is presented as follows:

$$sw_i(x) = \sum_{k=1}^{K} w_k * \hat{p}_k(i|x), \qquad i = 1, 2, \ldots, M, \tag{7}$$

where $w_k = 1 - Pe_{\Psi_k}$, and Pe_{Ψ_k} is the empirical error of Ψ_k classifier estimated on the testing set. In the case when the error is estimated on the learning set, we can talk about the estimation error based on the resubstitution method. Then w_k weight of each component classifier is calculated depending on the:

$$w_k = \frac{\sum_{n=1}^{N} I(\Psi_k(x_n) = i, j_n = i)}{N}. \tag{8}$$

The N value refers to the number of the learning set observations, which is used for estimating classifiers' weights, and j_n is the class number of the object with n index.

The obtained weights are normalized according to the formula:

$$\sum_{k=1}^{K} w_k = 1, \tag{9}$$

which means that the sum of weights of all classifiers from the ensemble is equal to unity. In this case, the final decision of the ensemble of classifiers is the following:

$$\Psi_{wSum}(x) = \arg\max_i sw_i(x). \tag{10}$$

Within the basic version (1) we have $w_k = 1$ for all $k = 1, \ldots, K$. Having defined the weight (8) you can easily use them for other rules (5) and (6).

Another approach to obtain weights is the calculation in each class separately. Then the corresponding weight is calculated from the equation:

$$w_{ki} = \frac{\sum_{n=1}^{N} I(\Psi_k(x_n) = i, j_n = i)}{\sum_{n=1}^{N} I(j_n = i)}. \tag{11}$$

Decision rules are clear, they are multiplied by the weights (11), and the sum method assumes designation of $\Psi_{wcSum}(x)$.

3 Static Classifier Selection

We will suggest now the method for determining weights for the individual base classifiers. The values of these weights are the basis for the selection of classifiers. These weights can be seen in the context of the interval logic. It means that the particular weights will not be provided precisely but their lower and upper values will be used. Therefore, each w_k weight of K-component classifier will be represented by the upper $\overline{w}_k^{ss}$ and lower $\underline{w}_k^{ss}$ value.

Let us now present the method for calculating weights for individual classifiers in which the basis for the determination of upper and lower values will be the correction classification of component classifiers. Having at the disposal a group of K component classifiers $\Psi_1, \Psi_2, \ldots, \Psi_K$ we ascribe at the learning set the probability of the correction classification Pc_{Ψ_k} for each of them. The upper value $\overline{w}_k^{ss}$ of k-classifier weight refers to the situation in which k-classifier was correct, while the other committee classifiers proved the correct prediction. The lower value $\underline{w}_k^{ss}$ describes the situation in which k-classifier made errors, while the other committee classifiers did not make any errors. The upper value is obtained from the dependence:

$$\overline{w}_k^{ss} = \frac{\sum_{n=1}^{N} UC_k^{ss_n}}{\arg\max_{l \in \mathcal{K}} \sum_{n=1}^{N} UC_l^{ss_n}}, \tag{12}$$

where

$$UC_k^{ss_n} = \frac{I(\Psi_k(x_n) = j_n)}{\sum_{l=1, l \neq k}^{K} I(\Psi_l(x_n) = j_n)}. \tag{13}$$

However, the lower value is obtained from the dependence:

$$\underline{w}_k^{ss} = \frac{\sum_{n=1}^{N} LC_k^{ss_n}}{\arg\max_{l \in \mathcal{K}} \sum_{n=1}^{N} LC_l^{ss_n}}, \tag{14}$$

where

$$LC_k^{ss_n} = \frac{I(\Psi_k(x_n) \neq j_n)}{\sum_{l=1, l \neq k}^{K} I(\Psi_l(x_n) = j_n)}. \tag{15}$$

Similarly, as in Eq. (11), we can calculate the weighted in class appropriate lower and upper values.

3.1 Classifier Selection

Given K classifiers from the initial pool of classifiers now we select L, $L \leq K$ classifiers to the ensemble. The final decision is made on the basis of L classifiers. In the selection process, we set the value of L. Then choose from the available pool L classifiers with the largest coefficients w_k. In the set of all w_k, $k = 1, \ldots, K$ we find the w_k^L with the L-th largest value. Now we define the set $w^{SS} = \{w_k : w_k \geq w_k^L\}$. It means that we select from L best classifiers. The advantage of this method is that it is very cheap computationally [20].

First we create coefficient α_k^{SS} according to the formula:

$$\alpha_k^{SS} = \begin{cases} w_k & \text{if } w_k \in w^{SS} \\ 0 & \text{otherwise} \end{cases}. \tag{16}$$

If we use the sum method for the final combination of classifier outputs, then the score of the selected group of classifiers is the following:

$$s_i^{SS}(x) = \sum_{k=1}^{K} \alpha_k^{SS} * \hat{p}_k(i|x), \qquad i = 1, 2, \ldots, M. \tag{17}$$

The final decision of the selected group of classifiers is made according to the formula:

$$\Psi_{wS}^{SS}(x) = \arg\max_i s_i^{SS}(x). \tag{18}$$

Before making the final decision the coefficients α_1^{SS}, ..., α_K^{SS} are normalized to unity.

4 Experimental Studies

In the experiential research 12 datasets were tested. Ten data sets come from the UCI repository [7]. Two of them have random observations in accordance with a certain assumed distribution. One of them has objects generated according to the procedure [6], this is the so called banana distribution, the second one, instead, has random objects drawn in accordance with the procedure [10]—Highleyman distribution. In both cases, the a priori probability distribution of classes amounted to 0.5, and for each class 200 elements were drawn. The numbers of attributes, classes, and available examples of the investigated datasets are introduced in Table 1. In the study feature selection was not performed [9, 19]. The aim of the experiments was to compare the quality of classifications of the proposed static selection method algorithms with the oracle method, sum method, and base classifiers. The oracle strategy correctly classifies the test sample if any of the classifier from the ensemble predicts the correction label for this sample.

The research assumes that the group of classifiers is composed of seven elementary classifiers. Three of them work according to the $k-NN$ rule where the k parameter is from the set $k \in 3, 5, 7$. For the four remaining base classifiers the decision trees are used, with the number of branches denoted as two and the depth of the precision tree

Table 1 Description of datasets selected for the experiments

Data set	Example	Attribute	Class
Banana	400	2	2
Breast tissue	106	10	6
Dermatology	366	33	6
Glass identification	214	10	6
Haberman's survival	306	3	2
Highleyman	400	2	2
Ionosphere	351	34	2
Irys	150	4	3
Pima indians diabetes	768	8	2
Sonar (Mines vs. Rocks)	208	60	2
Vertebral column	310	6	3
Wine	178	13	3

Table 2 Classification error for base classifiers and oracle classifier

Data set	Ψ_1	Ψ_2	Ψ_3	Ψ_4	Ψ_5	Ψ_6	Ψ_7	Oracle
Banana	0.029	0.03	0.031	0.042	0.05	0.051	0.061	0.015
Breast	0.43	0.459	0.5	0.319	0.373	0.305	0.405	0.103
Dermat	0.125	0.137	0.152	0.078	0.109	0.07	0.068	0.003
Glass	0.343	0.387	0.388	0.362	0.383	0.316	0.406	0.096
Haber	0.254	0.261	0.261	0.313	0.309	0.306	0.3	0.127
Highl	0.079	0.086	0.086	0.082	0.085	0.096	0.091	0.037
Ionos	0.159	0.167	0.177	0.113	0.119	0.117	0.136	0.025
Irys	0.035	0.033	0.024	0.046	0.048	0.046	0.052	0.011
Pima	0.261	0.254	0.241	0.269	0.256	0.265	0.264	0.078
Sonar	0.197	0.208	0.222	0.306	0.308	0.302	0.366	0.031
Verteb	0.181	0.17	0.171	0.173	0.207	0.179	0.209	0.043
Wine	0.146	0.177	0.207	0.073	0.096	0.116	0.132	0.009

having at most six levels. In the decision-making nodes, the Gini index or entropy are used. The results are obtained via a ten-fold cross-validation method.

Table 2 presents the classification error for all base classifiers and for the oracle classifier.

Table 3 show the results of classification for the initial ensemble of classifiers $L = 7$ and results after the classifier selection process $L = 5$, $L = 3$. For all the datasets and classifiers presented in these tables.

The obtained results confirm the validity of the application for the proposed method for classifiers selection. Additionally, it can be confirmed that the method proposed in the work can improve the quality of the classification in comparison with the other methods used in the work.

Table 3 Classification error for classifiers with weights calculated as the fraction of correctly classified objects

Data set	L	Ψ_S	Ψ_P	Ψ_M	Ψ_{wS}^{SS}	Ψ_{wP}^{SS}	Ψ_{wM}^{SS}	Ψ_{wcS}^{SS}	Ψ_{wcP}^{SS}	Ψ_{wcM}^{SS}
	7	0.04	0.039	0.04	0.039	0.039	0.039	0.039	0.039	0.039
Banana	5				0.036	0.035	0.036	0.039	0.037	0.037
	3				0.032	0.03	0.032	0.033	0.039	0.032
	7	0.319	0.37	0.319	0.311	0.37	0.311	0.319	0.37	0.319
Breast	5				0.286	0.368	0.286	0.314	0.365	0.319
	3				0.278	0.327	0.278	0.327	0.405	0.335
	7	0.051	0.08	0.051	0.051	0.08	0.051	0.051	0.08	0.051
Dermat	5				0.055	0.08	0.055	0.053	0.077	0.056
	3				0.066	0.07	0.066	0.054	0.072	0.059
	7	0.284	0.342	0.284	0.283	0.342	0.283	0.281	0.342	0.283
Glass	5				0.287	0.325	0.287	0.268	0.336	0.287
	3				0.278	0.313	0.278	0.291	0.348	0.301
	7	0.28	0.275	0.28	0.28	0.275	0.28	0.282	0.275	0.282
Haber	5				0.287	0.28	0.287	0.272	0.274	0.27
	3				0.289	0.285	0.289	0.28	0.266	0.272
	7	0.083	0.078	0.083	0.082	0.077	0.082	0.082	0.077	0.082
Highl	5				0.08	0.077	0.08	0.075	0.081	0.078
	3				0.086	0.083	0.086	0.08	0.09	0.086
	7	0.089	0.085	0.089	0.088	0.085	0.088	0.088	0.085	0.088
Ionos	5				0.079	0.081	0.079	0.088	0.089	0.089
	3				0.101	0.095	0.101	0.095	0.094	0.097
	7	0.05	0.048	0.05	0.05	0.048	0.05	0.05	0.048	0.05
Irys	5				0.052	0.046	0.052	0.048	0.043	0.048
	3				0.041	0.037	0.041	0.052	0.05	0.048
	7	0.235	0.238	0.235	0.233	0.238	0.233	0.233	0.237	0.233
Pima	5				0.233	0.231	0.233	0.229	0.231	0.229
	3				0.238	0.235	0.238	0.233	0.238	0.233
	7	0.213	0.259	0.213	0.216	0.259	0.216	0.217	0.259	0.217
Sonar	5				0.248	0.259	0.248	0.253	0.258	0.253
	3				0.284	0.277	0.284	0.275	0.264	0.273
	7	0.152	0.16	0.152	0.153	0.16	0.153	0.152	0.16	0.152
Verteb	5				0.151	0.161	0.151	0.154	0.159	0.151
	3				0.161	0.161	0.161	0.155	0.166	0.16
	7	0.068	0.159	0.068	0.068	0.159	0.068	0.068	0.159	0.068
Wine	5				0.068	0.157	0.068	0.071	0.157	0.077
	3				0.077	0.127	0.077	0.079	0.132	0.08

5 Conclusion

This paper discusses static classifier selection from a pool of base classifiers. The stage of the selection process uses a learning phase of the classifiers ensemble, in which information is obtained about the correction classification of each base classifier. This information provides the basis to obtain the weights for each base classifier.

The advantage of EoC presented in the paper is its possibility to work in parallel and distributed environment. Only the process of calculating the weights of classifiers requires data from all other processes which are the base classifiers.

Experimental studies were carried out on the datasets available from the UCI repository. They show that using the proposed in the work classifier selection is a good way. For the selected method for calculating weights, we obtained improvement of the classification quality measured by average error.

Acknowledgments This work was supported by the Polish National Science Center under the grant no. DEC-2013/09/B/ST6/02264 and by the statutory funds of the Department of Systems and Computer Networks, Wroclaw University of Technology.

References

1. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer, Secaucus (2006)
2. Cavalin, P.R., Sabourin, R., Suen, C.Y.: Dynamic selection approaches for multiple classifier systems. Neural Comput. Appl. **22**(3–4), 673–688 (2013)
3. Cyganek, B.: One-class support vector ensembles for image segmentation and classification. J. Math. Imaging Vis. **42**(2–3), 103–117 (2012)
4. Didaci, L., Giacinto, G., Roli, F., Marciali, G.L.: A study on the performances of dynamic classifier selection based on local accuracy estimation. Pattern Recognition, **28**, 2188–2191, 11/2005 (2005)
5. dos Santos, E.M., Sabourin, R.: Classifier ensembles optimization guided by population oracle. In: IEEE Congress on Evolutionary Computation, pp. 693–698 (2011)
6. Duin, R., Juszczak, P., Paclik, P., Pekalska, E., de Ridder, D., Tax, D., Verzakov. S.: PR-Tools4.1, A Matlab Toolbox for Pattern Recognition. Delft University of Technology (2007)
7. Frank, A., Asuncion, A.: UCI machine learning repository Irvine CA (2010) http://archive.ics.uci.edu/ml
8. Giacinto, G., Roli, F.: An approach to the automatic design of multiple classifier systems. Pattern Recognit. Lett. **22**, 25–33 (2001)
9. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**, 1157–1182 (2003)
10. Highleyman, W.H.: The design and analysis of pattern recognition experiments. Bell Syst. Tech. J. **41**, 723–744 (1962)
11. Ho, T.K., Hull, J.J., Srihari, S.N.: Decision combination in multiple classifier systems. IEEE Trans. Pattern Anal. Mach. Intell. **16**(1), 66–75 (1994)
12. Jackowski, K., Krawczyk, B., Woźniak, M.: Improved adaptive splitting and selection: the hybrid training method of a classifier based on a feature space partitioning. Int. J. Neural Syst. **24**(03) (2014)

13. Jackowski, K., Wozniak, M.: Method of classifier selection using the genetic approach. Expert Syst. **27**(2), 114–128 (2010)
14. Kittler, J., Alkoot, F.M.: Sum versus vote fusion in multiple classifier systems. IEEE Trans. Pattern Anal. Mach. Intell. **25**(1), 110–115 (2003)
15. Kuncheva, L.I.: A theoretical study on six classifier fusion strategies. IEEE Trans. Pattern Anal. Mach. Intell. **24**(2), 281–286 (2002)
16. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley New York (2014)
17. Lam, L., Suen, C.Y.: Application of majority voting to pattern recognition: an analysis of its behavior and performance. IEEE Trans. Syst. Man, Cybern, Part A **27**(5), 553–568 (1997)
18. Ranawana, R., Palade, V.: Multi-classifier systems: review and a roadmap for developers. Int. J. Hybrid Intell. Syst. **3**(1), 35–61 (2006)
19. Rejer, I.: Genetic algorithms in EEG feature selection for the classification of movements of the left and right hand. In: Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013, pp. 579–589. Springer (2013)
20. Ruta, D., Gabrys, B.: Classifier selection for majority voting. Inf. Fusion **6**(1), 63–81 (2005)
21. Smetek, M., Trawinski, B.: Selection of heterogeneous fuzzy model ensembles using self-adaptive genetic algorithms. New Gener. Comput. **29**(3), 309–327 (2011)
22. Suen, C.Y., Legault, R., Nadal, C.P., Cheriet, M., Lam, L.: Building a new generation of handwriting recognition systems. Pattern Recognit. Lett. **14**(4), 303–315 (1993)
23. Trawinski, K., Cordon, O., Quirin, A.: A study on the use of multiobjective genetic algorithms for classifier selection in Furia-based fuzzy multiclassifiers. Int. J. Comput. Intell. Syst. **5**(2), 231–253 (2012)
24. Ulas, A., Semerci, M., Yildiz, O.T., Alpaydin, E.: Incremental construction of classifier and discriminant ensembles. Inf. Sci. **179**(9), 1298–1318 (2009)
25. Woloszynski, T., Kurzynski, M.: A probabilistic model of classifier competence for dynamic ensemble selection. Pattern Recognit. **44**(10–11), 2656–2668 (2011)

E-book Reader Application Driven by Commands Provided by a Brain-Computer Interface Based on Motor Imagery Paradigm

Izabela Rejer and Andrzej Klimek

Abstract In order to implement a computer application that will be driven by commands provided by a brain-computer interface, a change in the control approach is necessary. In this paper we discuss the brain potentials used in the brain-computer interfaces in terms of their usefulness for controlling different types of applications. Our primary goal is to propose a simple control approach based on a motor imagery strategy and to present an example of a real computer application (an e-book reader) that applies this approach.

Keywords Brain-computer interface · BCI · E-book reader · Computer application driven by BCI · Brain potential · Motor imagery

1 Introduction

According to the formal definition, given by Wolpaw et al. [15], a brain-computer interface (BCI) is a communication system in which messages or commands sent by the user to the external world do not pass through the brain's normal output pathways of peripheral nerves and muscles. Instead, they are encoded in the brain activity and are propagated to the external world; they have to be read directly from the brain via a dedicated device. The device that is nowadays usually used for recording a brain activity is the electroencephalograph (EEG). In order to make use of brain signals recorded via the EEG device, the signals first have to undergo some mathematic transformations aimed at classifying them to one of the previously defined classes of brain activity patterns. Four main groups of algorithms are used in this process: signal preprocessing algorithms [11, 14], algorithms for feature extraction

I. Rejer (✉) · A. Klimek
West Pomeranian University of Technology, Szczecin, Żołnierska 49,
71-210 Szczecin, Poland
e-mail: irejer@wi.zut.edu.pl

A. Klimek
e-mail: aklimek@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_9

[13, 16], algorithms for feature selection [1, 6, 12], and classification algorithms [10].

There are a number of scientific papers that discuss different methods for processing EEG signals, however, relatively few of them touch the problem of designing applications dedicated to BCI users. Usually, the user application provides only a kind of simple feedback that helps the user to learn how to use the interface effectively. More sophisticated applications that are sometimes described in the scientific papers are aimed at writing text messages with use of evoked potentials, controlling the output devices with use of motor imagery paradigm, or at controlling game environment [5]. Meanwhile, most classic computer applications used in everyday life might be rewritten in such a way as to respond to the commands provided by a BCI. The only thing that is needed here is a slight change in the control approach.

The aim of this paper is to present an approach that can be used for controlling a computer application by commands provided by a BCI. The paper not only discusses the proposed approach but also presents how to apply it in a real computer application (an e-book reader). The proposed control approach is based on motor rhythms. Motor rhythms are brain potentials related to motor actions, such as arm, hand, or leg movements. They are spontaneous signals, which means that they do not need any external help to be generated. They appear in a user brain activity recorded over the motor cortex as a result of real or imagery movements of different body parts. Since the spatial resolution of EEG recording is rather low, it is difficult to distinguish between the activation of the closely neighboring areas of the motor cortex. Therefore, in order to ensure the high discrimination of the commands provided by a BCI, only two signals are assumed in the proposed approach—one corresponding to the imagery movements of the left hand and the other corresponding to the imagery movements of the right hand. The cortical areas responsible for real and imagery movements of both hands are located in distant parts of the motor cortex and hence it is relatively easy to detect their activation with a surface EEG.

2 Brain Potentials Used to Drive a BCI

The signals that are used nowadays to control the brain-computer interface can be divided into two main categories: evoked potentials and spontaneous signals. While evoked potentials are generated unconsciously by the subject when he perceives a specific external stimulus, spontaneous potentials are voluntarily generated by the user, without any external stimulation.

The main advantage of the evoked potentials (EP) is that the user does not need to be trained to use them for driving a BCI. As such, they can be used effectively from the first contact with the interface. On the other hand, the main disadvantage of EP is that they need external stimuli to be generated. Moreover, to detect some of them (e.g., P300 potential) the user has to be focused all the time on the fast and repetitive stimuli, which can be tiring and uncomfortable. There are two main types of EPs used for driving a BCI: Steady State Evoked Potentials (SSEP) and P300.

SSEP are brain potentials that appear when a subject perceives a periodic stimulus such as a light flickering at a given frequency. SSEP are defined as an increase in the EEG signal power in frequency being equal to the stimulation frequency or being equal to its harmonics and/or sub-harmonics [8]. Different types of SSEP have been used for driving a BCI, such as steady state visual evoked potentials (SSVEP) [7], somatosensory SSSEP [9], or auditory SSAEP [4]. SSEP appear in the brain areas corresponding to the sense that is being stimulated. Hence, SSVEP appear in visual areas in the occipital cortex, SSAEP in auditory areas in the temporal cortex, and SSSEP in sensory areas in the parietal cortex.

How does a BCI application based on SSEP work? Since the overall scheme is very similar for all types of SSEP, we present it on the example of only one of them—the visual SSEP. The scheme is very simple. A set of flickering objects is displayed in different positions on the screen, where each object has a unique flickering frequency. The user observes the screen and draws his attention to the object that corresponds to the action he wants to perform. When he focuses his attention on the object, the component of the frequency corresponding to the flickering frequency of this object, measured over the visual cortex can be detected in EEG signal. After detecting in EEG signal a component of a frequency f, the action attached to the object flickering at the frequency f, is performed. In this way it is possible to use the screen with flickering letters to write text messages.

The main advantage of SSVP is that they can be used to drive a BCI-based application without previous training, as all evoked potentials. Moreover, since periodic stimuli of different stimulation frequencies will lead to components of different frequencies, it is possible to use a large number of stimuli in order to obtain and use a large number of different brain activity patterns (even 48 stimuli were successfully tested in scientific research on SSVEP [3]). This enables to code a large number of commands which makes the whole system more convenient. Hence, on the one hand this paradigm is very promising. On the other hand, however, it has one serious limitation. It cannot be used on its own, but needs an external device for displaying symbols or emitting other types of stimuli, for example, acoustic or haptic.

The second type of evoked potentials that is often used to drive a BCI is a P300 potential. The P300 is a positive potential that appears over the parietal cortex when the user perceives a rare and significant stimulus. The P300 potential can be detected in the brain activity approximately 300 ms after the stimulus. The P300 potential is evoked with use of the so-called *odd-ball* paradigm. According to this paradigm, two kinds of stimuli are presented to the user. One of these stimuli appears very often, the other is rare. The user focuses his attention on the rare stimulus and as a result the P300 potential appears in his brain activity every time this rare and important stimulus is presented.

In P300-based BCI applications, a set of objects is displayed on the screen [2]. The objects are randomly highlighted and the user has to count the number of times that the object he is interested in is highlighted. In this way both assumptions for the odd-ball paradigm are fulfilled: the stimulus is rare, because it is one object out of the whole set and the stimulus is significant for the user because he has to pay attention to it (he has to count it) every time it appears. Why does the user have to count the stimuli? The reason is that the P300 potential is so small that it is difficult to be detected in a single trial experiment. That is why the same stimulus is presented to the user more than once. After each repetition the signals are averaged and after some iterations the potential P300 is large enough to be detected.

The second category of brain potentials that is often used for driving a BCI is a category of spontaneous signals. This category comprises three different types of signals: motor rhythms, slow cortical potentials, and non-motor cognitive signals. Motor rhythms are brain rhythms that appear in the brain activity in relation to motor actions (e.g., hand movement). Usually two motor rhythms are considered, rhythm μ (8–13 Hz) and rhythm β (13–30 Hz), both recorded over the motor cortex. Two different strategies have been proposed in order to enable the BCI user to control these motor rhythms: operant conditioning and motor imagery. In the first strategy, a user learns to modify voluntarily the amplitude of his motor rhythms through a long training. The main drawback of this strategy is that the training can last several weeks or even several months. However, when the training is completed, a stable performance can be obtained. The second strategy used to control a BCI with use of the motor rhythms is a strategy based on motor imagery. The outline of this strategy is as follows. A user imagines (or performs) movements of different body parts (hands, legs, arms, etc.). The brain potentials that appear as a result of performing or only imagining movements of different body parts have specific temporal, spectral, and spatial features. For instance, when the user imagines a left-hand movement, a decrease in power (Event Related Desynchronization (ERD)) of the μ and β rhythm over the right motor cortex can be observed. The same happens when the user imagines a right-hand movement but this time a relevant field in the left motor cortex is involved. These spatial, temporal, and frequency features are then used to decide on the part of the body in which movement was imagined or performed by a BCI user.

The main advantage of the motor rhythms is that they do not need any external stimulation to be generated. They can be generated voluntarily by the user at any time. Moreover, they enable the user to perform other actions during the BCI session. This means that the interface can be in an *on* state all the time but the user can use it only at some chosen moments. Of course, theoretically also the BCI based on P300 and SSVP could be constantly *on*, however, this would be extremely tiring and uncomfortable for the user. Hence, motor rhythms are the most convenient for the user using the BCI application. Moreover, if the motor imagery strategy is used to generate the motor rhythms, almost no training is needed. There is no free lunch, however. The motor rhythms have one serious drawback—they enable to differentiate only a very small number of commands. In a BCI based on P300 potential or SSVP EEG data are recorded from the same brain areas for different commands generated by the interface. The decision on the command chosen by the user is taken on the basis of the time when the potential was found in the EEG signal (in the case of P300 potential) or on the basis of the dominant frequency found in this signal (in the case of SSVP). Hence, the spatial resolution of EEG data is not so important when evoked potentials are considered. This spatial resolution of EEG data is, however, very important in the case of motor rhythms. Now the decision on the command that was chosen by the user is taken after analyzing in which areas of the motor cortex activation or deactivation of motor rhythms was observed. Since the sensors recording the EEG signals are far away from the real signal sources, the spatial resolution of the signal on the surface of the scalp is very low. As a result it is difficult to select the right source of the signal for the cortex areas that are close to each other. Therefore, BCIs based on the motor rhythm use data recorded only from distant areas of the motor cortex and as a result provide a very small number of commands. Very often in BCI research data from only several EEG channels are recorded.

Two last potentials from the group of spontaneous potentials that can be used to drive a BCI are slow cortical potentials and non-motor rhythms. Slow cortical potentials (SCP) are very slow variations of the cortical activity. It is possible to learn to make these variations positive or negative using operant conditioning. Non-motor rhythms are potentials trigger by non-motor cognitive tasks, such as mental mathematical computations, mental rotation of geometric figures, mental generation of words, etc. Both types of potentials are nowadays rarely used in real applications.

All potentials described so far have their advantages and disadvantages. The evoked potentials can be used without subject training but they require external stimulation and can be tiring for the user. Spontaneous signals are more natural and comfortable for the user, since they do not rely on external stimulation, but generally require a long training time. Among all the mentioned potentials, the most comfortable for the user are potentials triggered by motor imagery which have most of the advantages characteristic of spontaneous potentials and which do not have their main drawback, that is, they do not need long training time. In fact, in suitable conditions they do not need training at all. Their only drawback is the small number of commands that can be used to control an application.

3 Control Approach

A huge number of different control states can be distinguished when a computer application is controlled by a mouse and a keyboard. Since each state can be assigned to another command, a user of this application can easily move in a graphic environment or simply write texts. Everyday applications make use of the full potential of this concept, and are able to implement a large variety of dedicated functionality. But when it comes to software controlled by BCI, due to classification difficulty and user convenience, interfaces are forced to emphasize on simplicity.

Different types of applications driven by commands delivered by a BCI system have different requirements and hence need appropriate BCI paradigm. For example, in the case of applications oriented toward generating text, such as virtual keyboards or documents creators, P300 or SSVP paradigm are the better options. On the other hand, in the case of applications that need only occasional user interventions, the motor rhythms should be applied. E-book readers belong to the second group of applications. Their main task is to display a book or a document for a user. The user who works with the e-book reader spends most of the time simply reading the document. The control commands are provided by a user only occasionally in different time moments. Therefore, the control approach used in the e-book reader should be based on motor rhythms and apply the motion imaginary strategy. This strategy ensures that the system can be exploited without the necessity of prolonged user training.

Because of low spatial resolution of EEG signals, discussed in the previous section, the proposed approach will be based only on two different mental activity patterns. One of them will be generated when the user imagines movements of the left hand, the other when the right hand movements are imagined. Therefore, only two EEG channels will be used for EEG signal recording—C3 and C4. The signals acquired via both channels will be processed inside the BCI system and the information about the recognized activity pattern will be returned as an output. Since only one state will be recognized by the system at a time, only two commands corresponding to the imagery movement of the left and right hands will be delivered to the application.

Controlling the complex application with only two discrete states seems to be a difficult task. However, the authors propose a simple scheme that can be used to deal with this task. According to this scheme one state will be responsible for activating a selected option, and the other for changing the active selection. The best way to illustrate this concept is to imagine a finite set of operations executable on a running application. These operations implement the necessary functionality required from an e-book reader. The activation of an element from the set changes the state of the application, e.g., changes the page of a document that is displayed in the reader. The set must contain a position mark that informs which element will be executed when the approval command arrives. The position mark changes its value from the top to the bottom of the list of elements, when the second command is delivered. Since only two input signals are available, the set of elements must be traversed in a cycle to ensure that all options from the list can be reached by the user at any given time.

Introducing all the functionality implemented in the software to a single list would be extremely inefficient due to the time needed to select a required element. Additionally, adding a new functionality would mercilessly extend that time, leading to scalability restrictions. Another reason against the single list concept is that a lot of options displayed at the same time would confuse the user. Therefore, it is better to group operations into sublists using graphic components from the user interface as the grouping criterion. With such an approach each graphic component is bound to its own, often very specific, list of operations that can be executed by a user. Another advantage of the component specific list concept is the ability to maintain encapsulation—the control module can decide on the components that are available during different operations.

As a result of the component specific list concept, the proposed control approach has to be equipped with a mechanism for changing the focus of the user's input. This mechanism must provide methods to change active component, and must be available at any time, chosen by the user. When the active component is changed, the operations that are bound to the deactivated component should be deleted from the list of the available operations, and the interaction channel with the component should be broken. Until the component is active again, the user should not be allowed to change its state. At the same time, another component chosen by the control module should register its operations on the list of available operations.

4 E-book Reader

The implemented e-book reader application is divided into separate graphic panels that provide interactive functionality essential for the user. Thanks to the event-driven approach, each of these components can be reused on various occasions, e.g., file selection panel is instantiated twice, both times with a different purpose. External and internal events triggered outside all panels are handled by a dedicated communication channel linked to all the necessary components. There are many situations where the internal events are essential, for example a panel that renders page must be aware when a file is chosen, and must know the full content of the file path. This example shows that events not only inform panels when something happens, but also provide all the data necessary for describing the current situation. A user's input is also a type of an event—it contains information about the command chosen by the user.

The element that processes user's input, lists possible operations executable on an active panel and enables to shift focus between panels is *Toolbar*. *Toolbar* is always anchored to the bottom of the allotted screen space and is also always visible. Naturally, *Toolbar* does not contain information about all the operations available for the active component, but only provides a way to register an operation, and its visual representative (text button, icon, etc.). When the user picks one of the available options, *Toolbar* sends the information to the component that registered the operation bound to this option and asks this component for processing the request. The graphic element representing the option that is currently selected on *Toolbar* is distinctly larger than the elements representing other options, so that the user precisely knows what will happen when he sends an approval command (Fig. 1).

One of the application's requirements is that it must be driven by commands delivered from BCI right after startup. Hence, not only the e-book navigation mechanisms and the page manipulation system had to be considered when implementing the application, but also the ability to choose a file in the host file system and terminate the application at any given time. To deal with these tasks a menu panel has been brought to life. Its main purpose is to open a panel where file selection can be achieved, and to close the application when necessary; it is also the first panel presented to the user. Due to the menu's iterative structure, and the lack of complex operations (only two options exist in *Toolbar* at the beginning), one can wonder why not redirect input straight to the component rather than keep sending signals through another control layer (*Toolbar*). There are two main reasons against such an approach. First, using two different control approaches (one for menu and the other for the remaining operations) would be confusing for the user. Second, when the first component is being initialized, a new option that allows changing the focus to the previously visited panel is permanently added to *Toolbar*. This means that

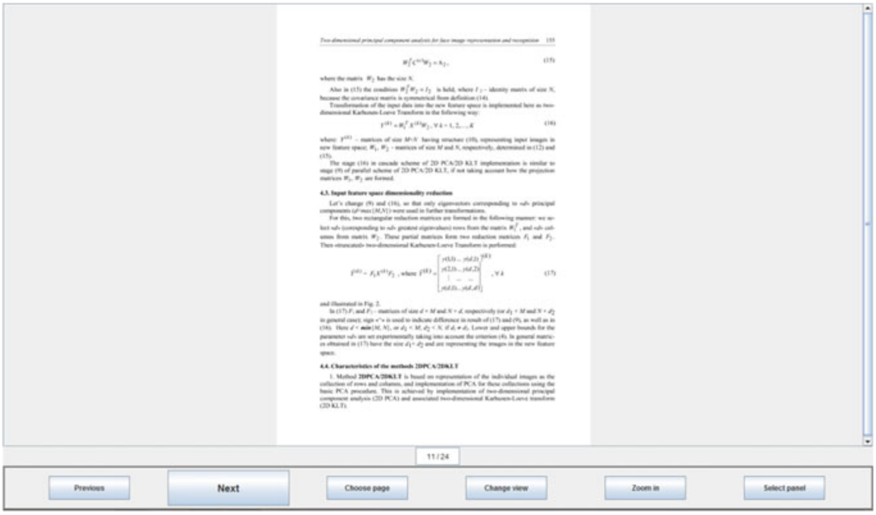


Fig. 1 E-book reader, *Toolbar* concept

instead of the two options available in *Toolbar* at the beginning, three of them are now present.

When the option *Open file* from the menu has been selected, the file selection panel is created and the view changes. The new panel consists of icons representing different system file elements such as directories, files, and drives. It is worth mentioning that in order to reduce the number of elements, only files of suitable extensions are presented in the view. Currently selected element is highlighted by a colored border. When icons occupy more space than the current window can offer—a scrollbar is added to the side. At the first program use, or when the data file is loaded incorrectly, the *File Selection* panel lists all drives available to the user. After the file has been chosen and the confirmation from the user has been granted, the current location is saved in the dedicated data file. Therefore, the next time the *File Selection* panel is visited, the previously saved location is displayed.

Except from the last visited location, also the file paths of the recently opened e-books are saved between the application runs. When at least one file was previously opened, the button *Last viewed* on the menu is turned on to indicate that a new panel can now be reached. The functionality of this new panel is almost the same as in the case of the *File Selection* panel. There are only two differences. First, *Last viewed* panel contains only files that the user had opened previously. Second, there is no possibility to travel within system file—all the interesting files are in one space/folder. When the user finally confirms the file selection, the main panel is created. This panel is responsible for rendering e-book pages to the user. In a default view, one page at a time is rendered, however, by choosing *Page mode* option, the user can switch between this default view and the view composed of two or four pages, all with a centered layout.

By default, rendered pages are shown at a full scale, so that all of the content can be viewed by the user. This can be changed by using *Zoom* option. When this option is selected, the content is stretched horizontally and a side scrollbar appears. Additionally, new *Toolbar* options appear, the first for sliding the content to the next position and the second for zooming back to the original page mode. When the end of page is reached, the scrollbar's knob returns to the starting position. The zoom ability is available only for one or two pages mode.

Except the page(s) to page(s) type of navigation, the application provides a way to traverse greater lengths within a file. *Select page* option enables the user to move directly to the chosen page. When *Select page* option is selected by the user, a new subpanel that resembles a simple calculator is created. This new panel allows to enter, digit by digit, the appropriate page number (Fig. 2). Like the common calculator, the panel has buttons to enter, delete, and calculate (or in this case confirm) numbers. When the confirmation button is activated, and the chosen number is not out of range, the appropriate page is displayed, and the focus returns to the main panel.

When two or more panels have been initialized in the application, an additional permanent option *Change panel* is attached to *Toolbar*. This option is responsible for changing focus between panels. After activating this option, the user gains access to all of the created panels. The panels are presented one at a time and are ordered according to the time when they were visited. *Toolbar* now has only two options, *Next panel* (the option changes the panel displayed on the screen) and *Activate panel* (the option activates the panel that is displayed on the screen). The panels that are displayed on the screen are covered with a gray overlay to indicate that they are now inactive and no interaction is taking place. After selecting the option *Activate panel*, the chosen panel regains focus and all previously known operations are again available on *Toolbar*; the overlay is removed.

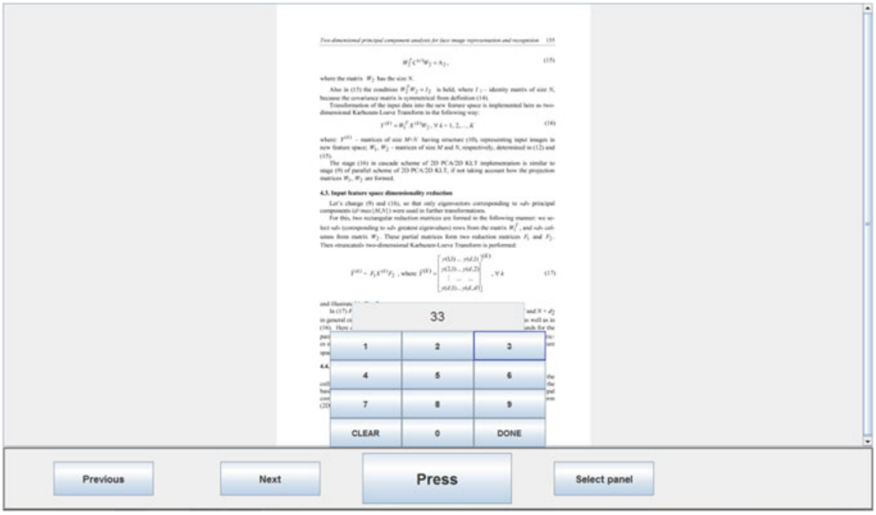


Fig. 2 E-book reader, *Select page* option

5 Conclusion

The e-book reader described in the previous section was implemented only a month before publishing this paper and hence it has not been tested yet with a target group; it is with disabled. However, the application was preliminary tested by the authors of this paper and also five other healthy subjects. At first, it was strange to get used to the concept of using the movements to one side (left) for moving around the list of choices and movements to the second side (right) for accepting the options, but after a few 15-minute sessions each of the seven testers was able to perform all the given tasks.

References

1. Burduk, R.: Recognition task with feature selection and weighted majority voting based on interval-valued fuzzy sets in computational collective intelligence. In: Nguyen, N.-T., Hoang, K., Jędrzejowicz, P., et al. (eds.) Technologies and Applications. Lecture Notes in Computer Science. Springer, Berlin (2012)
2. Donnerer, M., Steed, A.: Using a P300 brain-computer interface in an immersive virtual environment. Presence: Teleoper. Virtual Environ. **19**(1), 12–24 (2010)
3. Gao, X., Xu, D., Cheng, M., Gao, S.: A BCI-based environmental controller for the motion-disabled. IEEE Trans. Neural Syst. Rehabil. Eng. **11**(2), 137–140 (2003)
4. Hill, N.J., Lal, T.N., Bierig, K., Birbaumer, N., Schölkopf, B.: An auditory paradigm for brain-computer interfaces. Advances in Neural Information Processing Systems. Springer, Berlin (2005)
5. Holz, E.M., Höhne, J., Staiger-Sälzer, P., Tangermann, M., Kübler, A.: Brain-computer interface controlled gaming: Evaluation of usability by severely motor restricted end-users. Artif. Intell. Med. **59**(2), 111–120 (2013)
6. Koprinska, I.: Feature selection for brain-computer interfaces. In: Theeramunkong, T., et al. (eds.) PAKDD Workshops 2009. LNAI 5669, vol. 5669, pp. 100–111. Springer, Berlin (2010)
7. Lalor, E., Kelly, S.P., Finucane, C., Burke, R., Smith, R., Reilly, R., Mc-Darby, G.: Steady-state VEP-based brain computer interface control in an immersive 3-D gaming environment. EURASIP J. Appl. Signal Process. **19**, 3156–3164 (2005)
8. Lotte, F.: Study of electroencephalographic signal processing and classification techniques towards the use of brain-computer interfaces in virtual reality applications. PhD Thesis, INSA de Rennes (2009)
9. Muller-Putz, G.R., Scherer, R., Neuper, C., Pfurtscheller, G.: Steady-state somatosensory evoked potentials: suitable brain signals for brain-computer interfaces? IEEE Trans. Neural Syst. Rehabil. Eng. **14**, 30–37 (2006)
10. Pfurtscheller, G., Neuper, C., Schlögl, A., Lugger, K.: Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters. IEEE Trans. Rehabil. Eng. **6**(3), 316–325 (1998)
11. Rejer, I., Górski, P.: Independent component analysis for EEG data preprocessing—algorithms comparison. In: Proceedings of the 12th International Conference on Computer Information Systems and Industrial Management Applications CISIM 2013. Springer (2013)
12. Rejer, I.: genetic algorithms in EEG feature selection for the classification of movements of the left and right hand. In: Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013. Advances in Intelligent Systems and Computing, vol. 226, pp. 579–589. Springer (2013)

13. Sałabun, W.: Processing and spectral analysis of the raw EEG signal from the MindWave. Przegląd Elektrotechniczny **90**, 169–174 (2014)
14. Wang, Y., Shangkai, G., Xiaorong, G.: Common spatial pattern method for channel selection in motor imagery based brain-computer interface. In: Proceedings of the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference (2005)
15. Wolpaw, J.R., Birbaumer, N., McFarland, D.J., Pfurtscheller, G., Vaughan, T.M.: Brain-computer interfaces for communication and control. Clinical Neurophysiology, vol. 113(6), pp. 767–791. Elsevier (2002)
16. Yang, R., Song, A., Xu, B.: Feature extraction of motor imagery EEG based on wavelet transform and higher-order statistics. Int. J. Wavelets Multiresolut. Inf. Process. **8**(3), 373–384 (2010)

Comparison Between Mini-models Based on Multidimensional Polytopes and *k*-nearest Neighbor Method: Case Study of 4D and 5D Problems

Marcin Pietrzykowski

Abstract This paper presents the comparison between mini-models method based on multidimensional polytopes and *k*-nearest neighbor method. Both algorithms are similar, and both methods use samples only from the local neighborhood of the query point. The mini-models method can on the defined local area use any approximation algorithm to compute the model answer. The paper describes the learning technique of mini-models and presents the results of experiments that compare the effectiveness of two examined algorithms.

Keywords Mini-model · Local regression · K-nearest neighbors method · Mathematical modeling

1 Introduction

This paper presents a comparison between mini-models method [7–9, 11–13] based on multidimensional polytopes [10] and *k*-nearest neighbor method (*k*-NN) [2]. In contrast to most well-known methods of modeling such as neural networks, neuro-fuzzy networks, or polynomial approximation, the methods compared in this paper do not create a global model if it is not necessary. Mini-models, similar to the method of *k*-nearest neighbors, operate only on data from the local neighborhood of the query. These two methods are similar. The last method does not create any model $y = f(x_1, x_2, \ldots, x_n)$. The *k*-NN even can be considered as a particular kind of a mini-model. This is the main reason why the paper compares these two methods. Differences and similarities between them will be described in detail in a separate section in this paper. The minor aim of this paper is to describe learning algorithm of

M. Pietrzykowski (✉)
West Pomeranian University of Technology, Faculty of Computer Science
and Information Technology, Żołnierska 49, 71-210 Szczecin, Poland
e-mail: mpietrzykowski@wi.zut.edu.pl
http://www.wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_10

mini-models method in 4D- and 5D-space. Mini-models based on polytopes in 2D- and 3D-space were previously described in many publications [7–9, 11].

In the modeling task, we try to identify a mathematical function that describes the dependency between input and output variables. The approach of local modeling is a consequence of the fact that in the modeling process we are mostly interested in an answer for a specific query. Example of such query is: "What does y amounts to if x_1 amount to 0.5, x_2 to 0.8, x_3 to 0.1 and etc?" When a scientist is modeling air pollution on the road he can ask the question: "How large will the air pollution on a road be when the wind speed amounts to 5 m/s, temperature 2 m above the ground amounts to 24° C, number of cars per hour amounts to 200?" The answer to the question requires only the data "wind speed amounts about 5 m/s, temperature to 24° C and number of cars to 200." In the general case the query point is a set of independent variables. The query point will have here the following form: $x_1 = 5, x_2 = 24, x_3 = 200, y = ?$ or simply $x_1 = 5, x_2 = 24, x_3 = 200$.

2 Mini-models Method

The concept of the method of mini-models was developed by Piegat et al. [7, 8]. The method consists of two groups of algorithms: algorithms for defining the local neighborhood of the query point and algorithms for mathematical modeling on the mini-model area. The mini-model area can be defined as an area of a polytope placed in the general number of dimensions in the input space of the problem. It is a line segment in a 2D-space, polygon in 3D-space (triangle, quadrilateral, etc.), and polyhedron in 4D-space (simplex, hexahedron, etc.). In a n-dimensional space the mini-model area will be an n-1-dimensional convex polytope. The area could also have hyperellipsoidal shape [13]. In the initial researches concerning 2D- and 3D-space problem, the mini-model area was manipulated by changing the location of polygon vertices, but this approach has a lot of disadvantages in higher dimensional spaces. The most serious is "course of dimensionality." Hypercube in n-1-dimensional space contains 2^n vertices. Moreover, manipulation of a single point is impossible because it requires manipulation of other points belonging to the same face in order to maintain vertices coplanarity. Another problem is the task of including or excluding points within a mini-model area.

2.1 The Mini-models Area Embedded in the Spherical Coordinate System

The disadvantages of mini-models described above can be overcome by face manipulation in the coordinate system based on hypersphere [10]. This paper describes an algorithm for 4D and 5D problems. The method can be easily upgraded or downgraded to a required dimension. The first part of the algorithm is the data points conversion from Cartesian coordinate system into spherical coordinate system [1, 6, 14]. Transformation occurs only in the input space. The output variable remains untouched and is used in the process of a mathematical model identification. This means that 4D data points are converted into spherical coordinate system, but 5D data points into hyperspherical coordinate system based on 4D sphere. For ease of understanding, algorithm for 4D problems will be described first.

The query point Q will be the the center of the coordinate system. All data points p_i are transformed into spherical coordinate system and are defined by radius $r \in [0, \infty)$ (distance from the center) and two angles $\theta \in [0; \pi)$ and $\varphi \in [0; 2\pi)$. The transformations are obvious and can be found in any mathematical handbook. The set of points P can be denoted as:

$$\begin{aligned} P &= \{p_1, p_2, \ldots, p_i, \ldots, p_n\} \\ p_i &= (x_{i1}, x_{i2}, x_{i3}, y_i) = (r_i, \theta_i, \varphi_i, y_i). \end{aligned} \tag{1}$$

The mini-model area is in a 3D space polyhedron. Polyhedron contains J faces (for simplex $J = 4$, for cube $J = 6$, for octahedron $J = 8$). A particular polyhedron face j is a part of a plane F_j. In further considerations for the whole plane F_j we will call it simply *face*. A face is defined by a point G_j, which is called *face generation point*. There is an assumption that the plane is orthogonal to the vector $\overrightarrow{QG_j}$. Each face is defined as:

$$F_j = \left\{ G_j, p_i : \varphi_{ij} < \frac{\pi}{2} \wedge r_i = \frac{r_j}{\cos \varphi_{ij}} \right\} \tag{2}$$

where φ_{ij} is an angle value between vectors $\overrightarrow{QG_j}$, $\overrightarrow{Qp_i}$. The angle value can be computed using dot product, for Cartesian coordinates:

$$\varphi_{ij} = \arccos \frac{x_{i1}x_{j1} + x_{i2}x_{j2} + x_{i3}x_{j3}}{\sqrt{x_{i1}^2 + x_{i2}^2 + x_{i3}^2}\sqrt{x_{j1}^2 + x_{j2}^2 + x_{j3}^2}} \tag{3}$$

and for spherical coordinates we have:

$$\begin{gathered} \varphi_{ij} = \arccos((r_i \sin\theta_i \cos\varphi_i r_j \sin\theta_j \cos\varphi_j + r_i \sin\theta_i \sin\varphi_i r_j \sin\theta_j \sin\varphi_j + \\ r_i \cos\theta_i r_j \cos\theta_j)/r_i r_j) \end{gathered} \tag{4}$$

after simplification we obtain:

$$\begin{gathered} \varphi_{ij} = \arccos(\sin\theta_i \cos\varphi_i \sin\theta_j \cos\varphi_j + \sin\theta_i \sin\varphi_i \sin\theta_j \sin\varphi_j \\ + \cos\theta_i \cos\theta_j) \end{gathered} \tag{5}$$

and after using angle sum and difference formulas:

$$\varphi_{ij} = \arccos(\cos\theta_i \cos\theta_j + \sin\theta_i \sin\theta_j \cos(\varphi_i - \varphi_j)). \tag{6}$$

The face, in fact, divides all the space into two half-spaces. The first half-space consists of data points that may be included into the area of a mini-model. The set of points which may be included by face F_j is defined as:

$$I_j = \left\{ p_i : \varphi_{ij} \geq \frac{\pi}{2} \cup \left(\varphi_{ij} < \frac{\pi}{2} \wedge r_i \leq \frac{r_j}{\cos \varphi_{ij}} \right) \right\}. \tag{7}$$

The second half-space consists of data points that are certainly excluded from the figure area. The set of points certainly excluded by face F_j is defined as:

$$E_j = \left\{ p_i : \varphi_{ij} < \frac{\pi}{2} \wedge r_i > \frac{r_j}{\cos \varphi_{ij}} \right\}. \tag{8}$$

Every face divides the space in such way. Intersection of half-spaces (that include points) of all faces contains points that are included into the polyhedron area. Set of points Z included in the polyhedron are denoted as:

$$Z = I_1 \cap I_2 \cap \ldots I_J. \tag{9}$$

The way a face divides 3D space into two half-spaces is presented in Fig. 1. Data points marked by triangles are certainly excluded from the mini-model area, while points marked by squares are possibly included into the figure. Whether the point will be included in the mini-model area or not depends also on its position in relation to other faces. Only points included by all faces are included in the mini-model area. For the mini-model area defined in this way, any method of mathematical modeling can be used. The equations presented above refer to the polytope in 3D space; for a polytope in 4D space we have to replace the definition of p_i in (1) with:

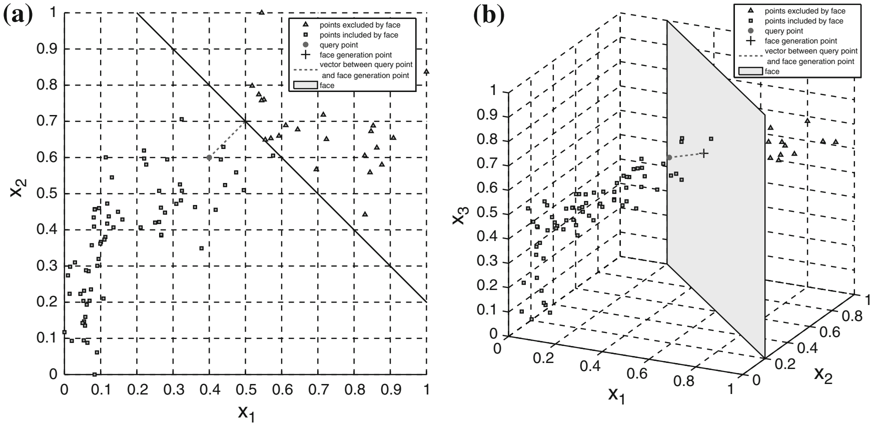


Fig. 1 Example of how the face divides input space of the problem into two half-spaces: in 2D-space projection (**a**) and in 3D-space (**b**)

$$p_i = (x_{i1}, x_{i2}, x_{i3}, x_{i4}, y_i) = (r_i, \varphi_{i1}, \varphi_{i2}, \varphi_{i3}, y_i) \tag{10}$$

where $\varphi_{i1}, \varphi_{i2} \in [0, \pi)$ and $\varphi_{i3} \in [0, 2\pi)$. After extending dimensionality, Eq. (3) has to be replaced with:

$$\varphi_{ij} = \arccos \frac{x_{i1}x_{j1} + x_{i2}x_{j2} + x_{i3}x_{j3} + x_{i4}x_{j4}}{\sqrt{x_{i1}^2 + x_{i2}^2 + x_{i3}^2 + x_{i4}^2}\sqrt{x_{j1}^2 + x_{j2}^2 + x_{j3}^2 + x_{j4}^2}} \tag{11}$$

and after all simplifications similar to above we obtain:

$$\begin{gathered}\varphi_{ij} = \arccos(\cos\varphi_{i1}\cos\varphi_{j1} + \sin\varphi_{i1}\sin\varphi_{j1}(\cos\varphi_{i2}\cos\varphi_{j2} + \\ \sin\varphi_{i2}\sin\varphi_{j2}\cos(\varphi_{i3} - \varphi_{j3}))).\end{gathered} \tag{12}$$

In the process of the mini-model learning the operation of figure rotation is very important [4]. This can be done by multiplying *faces generation points* coordinates by appropriate rotation matrix. In 3D space there are three rotation matrices:

$$R_{x_1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix} \; R_{x_2} = \begin{pmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{pmatrix} \tag{13}$$

$$R_{x_3} = \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

where ϕ is the rotation angle. Rotation in 4D-space is difficult to imagine because the first idea is to try to rotate about an axis in 4D-space. Rotation about an axis is an idea that comes from experience in 3D-space. Rotations in 3D-space should more properly be thought of not as rotations about an axis, but as rotations parallel to a 2D plane. In other words, we can write $R_{x_1} = R_{x_2x_3} = \cdots$, $R_{x_2} = R_{x_1x_3} = \cdots$, $R_{x_3} = R_{x_1x_2} = \cdots$. This way of thinking about rotations is consistent with 2D-space where only one plane exists. Thus, in 4D-space there are six rotation matrices:

$$
\begin{aligned}
R_{x_1x_2} &= \begin{pmatrix} \cos\phi & \sin\phi & 0 & 0 \\ -\sin\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} &
R_{x_1x_3} &= \begin{pmatrix} \cos\phi & 0 & -\sin\phi & 0 \\ 0 & 1 & 0 & 0 \\ \sin\phi & 0 & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
R_{x_1x_4} &= \begin{pmatrix} \cos\phi & 0 & 0 & \sin\phi \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\sin\phi & 0 & 0 & \cos\phi \end{pmatrix} &
R_{x_2x_3} &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\phi & \sin\phi & 0 \\ 0 & -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
R_{x_2x_4} &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\phi & 0 & -\sin\phi \\ 0 & 0 & 1 & 0 \\ 0 & \sin\phi & 0 & \cos\phi \end{pmatrix} &
R_{x_3x_4} &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\phi & -\sin\phi \\ 0 & 0 & \sin\phi & \cos\phi \end{pmatrix}
\end{aligned}
\tag{14}
$$

2.2 *Mini-model Learning*

The first part of a mini-model learning (finding mini-model area) is heuristic. It usually involves changing location of points G_j in a random way. It involves changing radius (distance between face and query point) and changing angular coordinates. There are numerous mini-model areas available. When a mini-model area has been defined, any method of mathematical modeling can be use, e.g., polynomial approximation, mean value, fuzzy reasoning. In the next part of the learning process, the method of modeling uses only points that are encircled in the mini-model area. It uses selected points to calculate the error committed by the model on the learning data and then calculates the numeric answer for the query point. Then the algorithm tries to find another mini-model area, calculates the error, the mini-model answer, etc. The model that commits the smallest error is chosen as the optimal one. Not every mini-model area is valid, it has to satisfy following initial properties:

- minimal number of points inside the mini-model area,
- maximal number of points inside the mini-model area,
- ratio between the minimal and the maximal lengths of vectors $\overrightarrow{QG_j}$,
- query point should not be extrapolated by learning points (it is not always possible and sometimes it is not required or necessary).

There is no simple answer for how to choose initial values of the properties. The minimal and the maximal number of points inside the mini-model area depends on learning data, but the lower boundary has to be greater than the dimensionality of the problem space. It is possible that, for a particular query point and specified initial properties, there exist no valid mini-model area. In such situation, the mini-model is unable to return a reliable numeric answer. Results of preliminary numerical experiments have shown that mini-models very often were overfitted when faces of a polytope were changed in a completely random way. The proposed method of learning includes rotation of the whole figure and changing radii of face generation points many times. Rotation angle, rotation direction, and radius delta were taken at random. The ratio between the minimal and the maximal radii values has to be greater than 0.5 in order to prevent the figure from overstretching. A simplified algorithm of mini-model learning is presented below.

Simplified mini-models method learning algorithm.

```
BEGIN
    FOR (p in P)
        normalize data point p to [0; 1];
        convert data point p to spherical coordinates;
    END
    WHILE (bestError <= acceptableError OR
           iteration < iterWithoutImprovment)

        rotationAngle = Random();
        rotationPlane = Random();
        radiusDelta = Random();

        FOR (i = rotationAngel; i < 360; i += rotationAngle)
            rotate polytope by rotationAngel about rotationPlane;
            /* radiusDelta can be negative */
            radiusDelta = Random();
            facesNumbers = Random();

            FOR (j = radiusDelta; Abs(j) < 0.5; j += radiusDelta)
                change radius of specified faces by radiusDelta;
                Z = calculate points pi included by polytope;
                IF Z meets initial criteria THEN
                    error = calculate mini-model error;
                    calculate mini-model answer to query point;
                    IF (bestError > error) THEN
                        bestError = error;
                        store mini-model as winning model;
                        iteration = 0;
                    END
                END
            END
            revert faces radiuses to initial values;
        END
        revert mini-model parameters to values
            of last winning model;
    END
END
```

This is a template of the algorithm and it can be modified in many ways. For example, two mini-model areas are considered to be equal when the set of learning points included by them are equal. This situation can be handled by using a hash table that stores the mini-models areas previously checked.

3 Differences and Similarities Between Mini-models and *k*-nearest neighbors method

The mini-models method is similar to the method of *k*-nearest neighbors. Both algorithms use data samples from the local neighborhood of the query point that is important for a scientist. The methods require relatively small number of data samples in the learning process. This fact has great importance in a situation of data deficiency that very often occurs in real problems. Both methods calculate the answer to the query point "ad-hocly", which allows them to work in situations where new data points are continuously being received. This approach is free from the time-consuming process of creating a global model. However, it is also possible to build a global model in order to learn the value of a modeled variable across the entire domain.

The *k*-NN method is evaluated by many scientists as very effective and some of them are of the opinion that other methods are not necessary [5]. During the calculation of an answer for a query point in *k*-NN only *k* nearest samples are taken into account. The base version of the method uses Euclidean metric to identify nearest samples. In the classic *k*-NN method, the model answer is calculated as the mean value of the target function values or the weighted (by distance) mean value. All *k* nearest points are encircled in the area of a circle in 3D-space problem, sphere in 4D-space problem, and hypersphere in a general number of dimensions. In contrast, in the mini-models method points are encircled in the area of any polygon in 3D-space problem, polyhedron in 4D-space problem, and polytope in general number of dimensions. The number of samples in the mini-models method is not constant but should be placed within specified range. It has such implication that for a particular query point, many local neighborhoods are possible. We can consider *k*-NN method as a particular kind of a mini-model method. Taking that criteria mini-model area in the *k*-NN is a *n-1*-dimensional hypersphere in a *n*-dimensional problem. The algorithm of modeling here is the simple mean value or the weighted (by distance) mean value. Only one mini-model area for a particular query point and specified *k* is possible. Figure 2a, b show differences in an input space, while Fig. 2a, b shows how methods perform in full space. Another important difference is that *k*-NN method use only the mean value, while mini-models take into account not only samples target values, but also a tendency in the neighborhood of the query point. Using the information about this tendency causes better modeling in areas with information gaps.

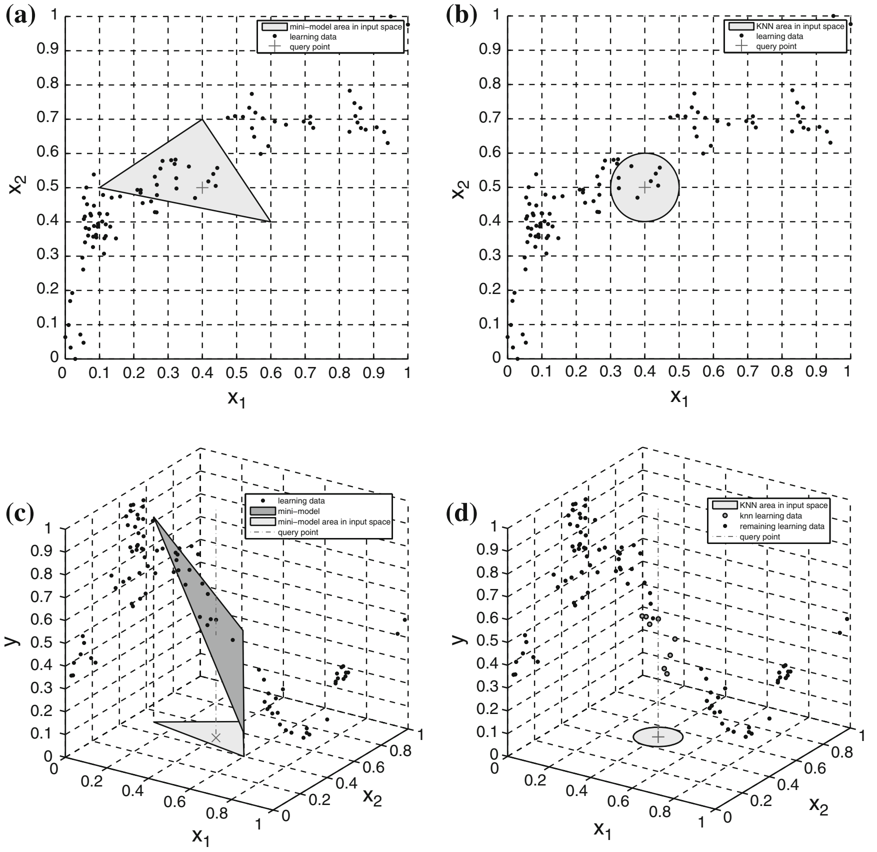


Fig. 2 Mini-model area (**a**) and *k*-NN (**b**) in input space (2D). Mini-models method (**c**) and *k*-NN method (**d**) in full space (3D)

4 Experiments

The experiments were conducted on three datasets:

- Unemployment rate in Poland [15] (x_1-number of employed, x_2-number of pensioners, x_3-money supply, x_4-dollar/zloty exchange rate, x_5-inflation rate, x_6-production, x_7-population, x_8-export value, x_9-import value) (96 instances),
- Concentration of NO_2 measured at Alnabru in Oslo [17] (x_1-number of cars per hour, x_2-temperature 2 m above the ground, x_3-wind speed, x_4-the temperature difference between 25 and 2 m above ground, x_5-wind direction, x_6-hour of day) (500 instances),
- Fuel consumption in miles per gallon [17] (x_1-number of cylinders, x_2-displacement, x_3-horsepower, x_4-weight, x_5-acceleration) (392 instances).

Table 1 Comparison of effectiveness of tested methods with dataset: unemployment rate in Poland

	k-NN		Mini-models method		
Input attributes	Error	*k* value	Error	Samples range	Base
2, 3	0.0294	2	0.0303	4–15	Square
3, 9	0.0776	4	0.0711	6–15	Square
2, 3, 7	0.0248	2	0.0251	5–15	3-simplex
3, 5, 9	0.0596	3	0.0600	8–15	Cube
2, 3, 5, 7	0.0252	2	0.0231	5–15	4-cube
1, 4, 6, 8	0.0977	3	0.1082	8–15	4-orthoplex

Table 2 Comparison of effectiveness of tested methods with dataset: concentration of NO_2 measured at Alnabru in Oslo

	k-NN		Mini-models method		
Input attributes	Error	*k* value	Error	Samples range	Base
3, 6	0.0921	27	0.1051	8–30	Triangle
1, 5	0.0935	27	0.1113	8–20	Square
1, 3, 5	0.0834	14	0.0946	9–30	Cube
3, 5, 6	0.0919	19	0.1017	9–30	Cube
1, 3, 5, 6	0.0843	13	0.0784	30–60	4-orthoplex
1, 2, 3, 5	0.0805	9	0.0827	15–80	4-orthoplex

The first dataset has a relatively small number of samples, the second is a dataset with noise. Experiments were performed with the optimal values for all parameters, for all tested methods. The experiments compare *k*-NN method with three variants of mini-model: based on simplex, cube, and orthoplex. The dimension of the polytopes was appropriate to the dimension of an input space, thus in 3D-space mini-models were based on triangle or square in 4D-space on tetrahedron, cube, or octahedron and in 5D on 4-simplex, 4-cube, or 4-orthoplex. Mini-models use single artificial neuron [3, 16] as a method of modeling. The method discarded the result for a particular query point if it was unable to find a valid mini-model area. The results of experiments have shown that in this situation usually error was very high.

Methods accuracy was measured with the use of leave-one-out cross-validation method. The method takes one point from the original dataset and uses it as a query point and the remaining points as learning data. Then the method calculates numeric answer for the query point, and the error committed by the model. After that, the point is returned to the whole set and another point is taken as a query point. This is repeated for each data point: each point in the original dataset is used once as the query point. After finishing the calculation, the mean error committed by the method was calculated. Tables 1, 2 and 3 present the results of experiments.

Table 3 Comparison of effectiveness of tested methods with dataset: fuel consumption in miles per gallon

	k-NN		Mini-models method		
Input attributes	Error	*k* value	Error	Samples range	Base
3, 5	0.0822	26	0.0862	8–40	Square
2, 4	0.0793	15	0.0734	8–40	Square
1, 2, 3	0.0734	15	0.0632	14–40	Simplex
3, 4, 5	0.0775	12	0.0815	14–40	Simplex
2, 3, 4, 5	0.0750	6	0.0670	15–100	4-orthoplex
1, 2, 3, 4	0.0730	9	0.0716	16–40	4-cube

5 Conclusion

First of all, the approximation function based on mini-models proved to have good accuracy. Mini-models perform slightly worse with datasets with noise (Table 2) but they have very advantageous extrapolation properties. It results from the fact that they take into account a tendency in the neighborhood of the query point. Using the information about this tendency causes better modeling in places with information gaps. A mini-model can detect extrapolation areas and situations in which initial criteria cannot be satisfied and thus the model is unable to return a reliable numeric answer. Generally, these are situations in which information gaps strongly interfere with the result of approximation. A certain weakness of mini-models in comparison with the *k*-NN method is the necessity of taking into account a greater number of points. Generally, *k*-NN method performs faster. The mini-models method is partially heuristic and sometimes its results may slightly vary. In this paper, only one variant of the method was tested: mini-model that used simple artificial neuron. In the next research, the author plans to check more complex variants of the method, e.g., where simple neural network will be used as a method for modeling on the mini-model area. The method also requires experiments on datasets with a higher number of attributes.

References

1. Bronshtein, I., Semendyayev, K., Musiol, G., Muhlig, H.: Handbook of Mathematics. Springer (2007). ISBN 9783540721215
2. Fix, E., Hodges, J.L.: Discriminatory analysis, nonparametric discrimination: Consistency properties. Randolph Field, pp. 1–21. Texas (1951)
3. Flasinski, M.: Wstep do Sztucznej Inteligencji. PWN, Warszawa (2011)
4. Hollash, S.R.: Four-space Visualization of 4d Objects. MSc. Thesis, Arizona State Univeristy (1991). http://steve.hollasch.net/thesis/
5. Kordos, M., Blachnik, M., Strzempa, D.: Do we need whatever more than k-NN? In: Proceedings of 10th International Conference on Artificial Intelligence and Soft Computing, Zakopane (2010)

6. Moon, P., Spencer, D.: Field Theory Handbook: Including Coordinate Systems, Differential Equations, and Their Solutions. Springer (1988). ISBN 9780387027326
7. Piegat, A., Wasikowska, B., Korzeń, M.: Application of the self-learning, 3-point mini-model for modelling of unemployment rate in Poland [in Polish]. Studia Informatica, University of Szczecin, No. 27, pp. 59–69 (2010)
8. Piegat, A., Wasikowska, B., Korzeń, M.: Differences between the method of mini-models and the k-nearest neighbors an example of modeling unemployment rate in Poland. In: Information Systems in Management IX-Business Intelligence and Knowledge Management. WULS Press, Warsaw (2011)
9. Pietrzykowski, M.: Comparison of effectiveness of linear mini-models with some methods of modelling. Młodzi Naukowcy dla Polskiej Nauki. CREATIVETIME, pp. 113–123. Krakw (2011)
10. Pietrzykowski, M.: Mini-models working in 3D space based on polar coordinate system. Nowe trendy w naukach inzynieryjnych 4. Tom II, CREATIVETIME, pp. 117–125. Krakw (2013)
11. Pietrzykowski, M.: Effectiveness of mini-models method when data modelling within a 2D-space in an information deficiency situation. J. Theor. Appl. Comput. Sci. **6**(3), 21–27 (2012)
12. Pluciński, M.: Mini-models—Local regression models for the function approximation learning. In: Rutkowski L., et al. (eds.) Proceedings of ICAISC 2012, Part II. LNCS, vol. 7268, pp. 160–167. Springer, Berlin (2012)
13. Pluciński, M.: Nonlinear ellipsoidal mini-models—application for the function approximation task. Przeglad Elektrotechniczny (Electrical Review), R. 88 NR 10b, pp. 247–251 (2012)
14. Polyanin, A., Manzhirov, A.: Handbook of Mathematics for Engineers and Scientists. Taylor & Francis (2010). ISBN 9781584885023
15. Rejer, I.: Metody modelowanie wielkowymiarowego systemu z użyciem metod sztucznej inteligencji na przykładzie bezrobocia w Polsce. Szczecin, Wydawnictwo Naukowe Uniwersytetu Szczecińskiego (2003)
16. Rutkowski, L.: Metody i techniki sztucznej inteligencji. PWN, Warszawa (2009)
17. UCI Machine LEARNING REPOSITORY: http://archive.ics.uci.edu/ml/

Part II
Design of Information and Multimedia Systems

Sushi: A Lightweight Distributed Image Storage System for Mobile and Web Services

Kamoliddin Mavlonov, Yoshinobu Higami and Shin-ya Kobayashi

Abstract This paper describes a lightweight image storage system, called Sushi, which has been designed for high traffic mobile and Web applications. The system aggregates the best practices in business and academic researches to achieve simplicity while providing high performance, availability, and scalability. The key design feature of the system is its use of an underlying nonblocking architecture with current software standards.

Keywords Image storage · Scalability · Distributed system · Content addressable

1 Introduction

Currently, we are seeing an explosion of new mobile and Web services (applications) due to the annual increase in the Internet usage [1]. Most services involve surfing online images. Among other challenges, high traffic services face image-surfing-related challenges (e.g., high lookup cost and reliability). The commonly applied practices fail to offer a good balance between complexity and performance. For instance, although image-caching in high traffic services improves performance, the solution is too complex for small to medium level service providers and noncommercial institutes.

Images need to be optimized in size to suit the mobile and Web environment. Moreover, they need to be secured in data stores for integrity and for easy and fast retrieval. In high traffic services, when the number of real-time concurrent

K. Mavlonov (✉) · Y. Higami · S. Kobayashi
Graduate School of Science and Engineering, Ehime University, 3 Bunkyou-cho,
Matsuyama, Ehime 790-8577, Japan
e-mail: kamol@koblab.cs.ehime-u.ac.jp

Y. Higami
e-mail: higami@cs.ehime-u.ac.jp

S. Kobayashi
e-mail: kob@cs.ehime-u.ac.jp

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_11

connections increases, I/O become very intensive and can easily degrade the overall performance. Sushi offers solutions to these challenges for small to medium level service providers with high traffic levels.

In the next section, we provide background information. In Sect. 3, we summarize solutions used to address the commonly encountered performance challenges which were highlighted in the Introduction. We also provide an overview of our proposed Sushi system. In Sect. 4, we present the design of Sushi. In Sect. 5, we present a sample of Sushi RESTFul API and we implement its design in Sect. 6. In Sect. 7, we evaluate Sushi for robustness and performance.

2 Background

In mobile and Web applications, there are a number of image-surfing challenges, which can be attributed to various issues. During upload/download of images to/from the Web server using HTTP, the socket connection remains open. Moreover, the socket connections still remain while the Web server receives and stores images into a persistent storage. The challenge for Web server solutions is still to be able to scale and keep receiving new connections without refusing or blocking them. Another main challenge of the existing solutions is the way of storing images into persistent storage and able to retrieve them fast despite the huge amount of images.

2.1 Nonblocking I/O

On the Web server side it is required to use asynchronous and nonblocking I/O solutions, which will fully utilize a server multicore CPU and disk sufficiently and are able to scale to tens of thousands of open connections for long polling, WebSockets that require a long open connection for each user.

2.2 Duplicate Files

Duplicate files can cause unnecessary utilization of resource like Disk Space, CPU, or even RAM. It is very common to have duplicate files in image storage systems for mobile/Web applications. To validate a hypothesis regarding duplicate files, we take three different business projects and classify them into three categories: Startup, Growing, and Big project, respectively. We find 7, 69, and 35 % of duplicate files in Startup, Growing, and Big project, respectively. Figures 1, 2, and 3 show a file and its duplicate amount in Small, Growing, and Big projects, respectively.

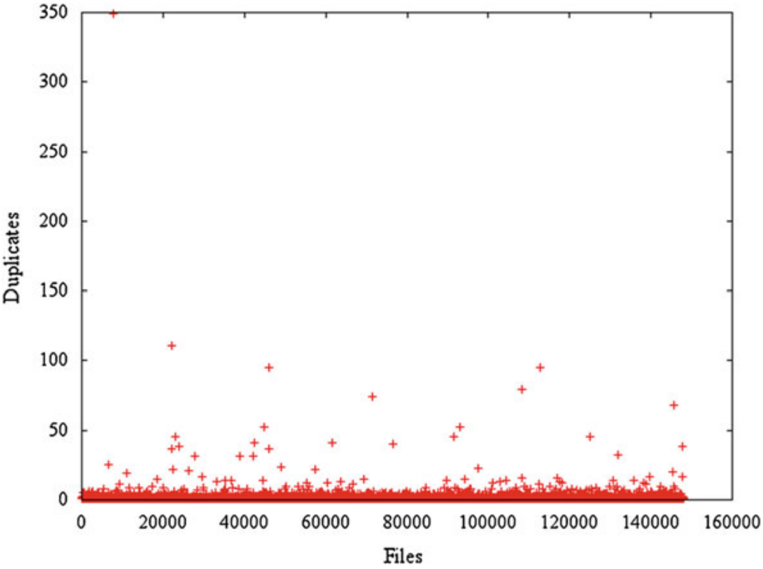


Fig. 1 Duplicate images in startup project

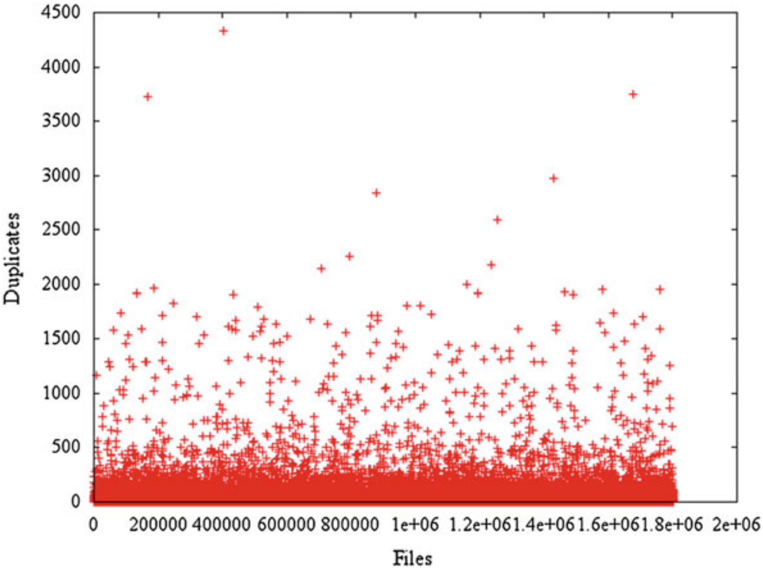


Fig. 2 Duplicate images in growing project

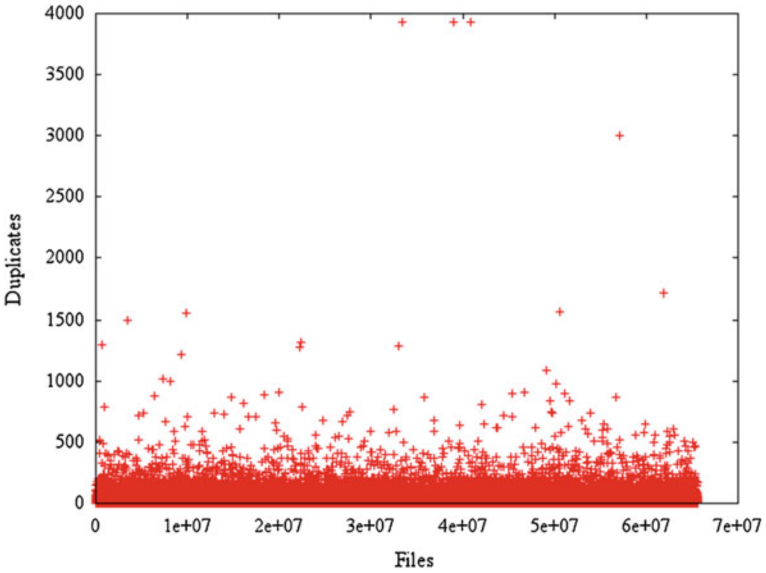


Fig. 3 Duplicate images in big project

2.3 Image-Related Feature Modules

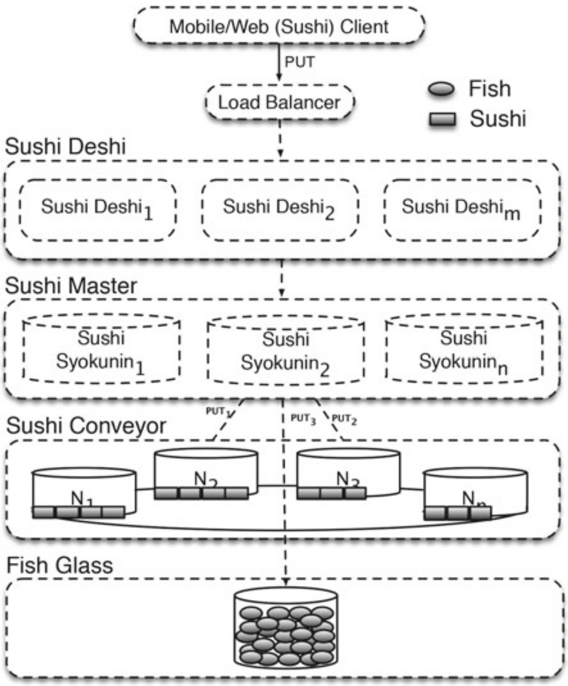
Knowing an image dimension before uploading an image itself is critical for Web/mobile (even native) applications. Thus, it allows the Web browser or mobile application to render or draw image placeholder in advance and efficiently. Moreover, faster rendering HTML page or drawing activity (in case of Android) improves user experience. To avoid this issue, developers have to implement their own solution to resize an image and store its width and height info in some way. The same issue is with image dominant color or image formatting or even handling the image's portrait or landscape mode.

3 Image-Surfing Approaches

(a) Traditional local file systems (LFS) with POSIX semantics organize directories in a flat, sequential data structure. This results in an $O(\mathrm{n})$ lookup cost which for large directories might cause performance bottleneck [2].

(b) Due to the I/O bottleneck, it is common to use caching solutions (e.g., Varnish [3] and Squid [4]) or make Web servers to cache static content (e.g., Nginx [5]) to speed up the image file delivery. However, it is a workaround, thus it will not solve the actual problem. Moreover, it introduces more complexity with caching mechanism and image synchronizations between cache nodes in real-time Web services.

Fig. 4 Big overview



The use of services like Content Delivery Network (CDN) is expensive [6], which does not align with our goals. (c) Sushi: This is a novel lightweight image storage optimization system. Different from most existing generic systems, Sushi has been designed specifically for optimizing image storage in Web and mobile services. Figure 4 shows the nonblocking architecture of Sushi consisting of the components (The Sushi Client, Sushi Deshi, Sushi Master, Sushi Conveyor, and Fish Glass) that will be explained in detail in the following section. This design characterizes Sushi in a way that enables it to achieve the many features that are desired by mobile and Web service providers:

- **Simplicity:** Simple as possible. Easy to integrate and deploy. Maintain and develop new features.
- **Robustness:** If any hardware failures or even entire server in a cluster is not available. Moreover, it should provide cross data center (DC)s or regional replication for disaster recovery (DR).
- **Modularity:** A loosely coupled distributed object storage system. All communication through standard API calls.
- **Portable:** Running on any Operating System with any file system.
- **High performance:** It should provide high performance and availability.
- **Scalability:** Seamless scaling in times of high traffic.
- **Secured:** It should be secure by default.

4 Design

Sushi consists of five major components: Sushi Client, Sushi Deshi, Sushi Master, Sushi Conveyor, and Fish Glass. These components, as shown in Fig. 4, take care of the whole life cycle of the image as defined in Sushi. Sushi's definition of image life cycle is inspired by fish and Sushi cycle.

4.1 Sushi Client

The main objective of Sushi Client is to provide an endpoint API for mobile and Web application. It is also an authentication bridge between an application and the Sushi Master. All communications are through standard HTTP/HTTPS protocol by Representational State Transfer (REST) API calls. For e.g., GET method of HTTP(S) protocol retrieves an image (bytes) from Sushi Conveyor directly by HTTP response (BODY), which can be cached by Web servers or proxies, according to REST protocol. The REST API samples of Sushi are shown in Sect. 5. However, PUT request is forwarded to a load balancer of Sushi, which is shown in Fig. 4.

4.2 Sushi Deshi

Sushi Deshis are daemons, which receive images by PUT in HTTP request (BODY), as shown in Fig. 4. Sushi Deshi is responsible for processing images, such as, receiving, resizing into different sizes, and generating dominant color and takes sum of SHA-1 [7] from content of the image and passes to Sushi Master for further processes. All requests are distributed by a load balancer between Sushi Deshis. When any of Sushi Deshi fails, the load balancer will stop delegating further requests for the failed Sushi Deshi.

4.3 Sushi Master

Sushi Master receives resized images by Sushi Deshi and stores in Sushi Conveyor. It stores the image into certain Sushi Nodes, which are chosen by consistent hashing (CH) algorithm. The main objective of Sushi Master is a reverse proxy functionality and population of CH. Moreover, the Sushi Master is coordinated by Raft protocol [8], to elect a leader. If a Sushi Master is faulty or fails, another Sushi Syokin (Master) will be elected automatically.

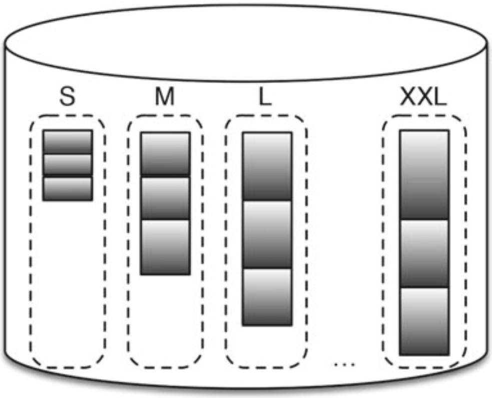


Fig. 5 Sushi Tubes in Sushi Node

4.4 Sushi Conveyor

Sushi Conveyor incorporates a consistent hashing algorithm and an implementation of consistent ring, which is similar to Cassandra [9] or original to Amazon Dynamo [10]. The ring encapsulates Sushi Nodes and Sushi Nodes consist of Sushi Tubes.

4.5 Sushi Tubes

Sushi Tubes contain log structured files (inspired by [6, 11]) as shown in Fig. 5. Sushies in Sushi Tubes are grouped by their sizes into respective log structured files. All I/O in log structure file (S/M/L group, etc.) are done with dedicated processes. Usually, in API calls only one-sized groups are requested. For e.g., only 10–20 thumbnails (S-sized group) or only 2–6 middle size images (M-sized group) per request. It is a characteristic of mobile and Web application [6]. Thus, these files are located close to each other or at least within one file (e.g., S-sized group), they can be located by just moving file offset to exact position and read sequentially. In this scenario, M or L groups are not touched.

4.6 Fish Glass

Fish Glass contains original image files (Fishes), as shown in Fig. 4. Fishes can be stored in any local file system (LFS), however, directory structure is required to be built with two-level hierarchy, inspired by Nginx [5]. Each level consists of directories from 00 to *FF* in hexadecimal range notation which can fit into one byte in hexadecimal and range from 0 to 255 in decimal notation. Fishes are placed into the directories according to the Sushi Id. For e.g., if the file checksum is $4dc8263cf28b66502d0d7581384e527e0000434a$, the first level of directory will be $4d$ and the second will be $c8$ accordingly. In total, 65,536 directories will be created in the process of Fish Glass initialization. This approach prevents large directories from appearing. The large directories can cause performance bottleneck [2] due to O(n) lookup cost.

Fish Glass itself is a distributed loosely coupled system, which can be replicated across DCs and regions. Moreover, Fish Glass is a service-oriented architecture [12], which can work without using Sushi Conveyor directly.

5 API

All communication between the Sushi and Web/mobile applications or Web services are done over standard HTTP(S) protocol with RESTFul API. It allows to decouple the Sushi as a service-oriented architecture. Examples of the PUT, DELETE, GET, and HEAD requests of HTTP protocol are shown below:

```
GET /[FID] HTTP/1.1
Host: www.example.com
RESPONSE:
Content-Type:image/jpeg
HTTP/1.1 200 OK
[FILE CONTENT]

HEAD /[FID] HTTP/1.1
Host: www.example.com
RESPONSE:
Content-Type:image/jpeg
HTTP/1.1 200 OK

PUT / HTTP/1.1
Host: www.example.com
Authorisation: SECRET KEY?
Content-Length: 65534
[file content]
RESPONSE:
HTTP/1.1 200 OK
[FID]

DELETE /[FID] HTTP/1.1
Host: www.example.com
Authorisation: SECRET KEY?
RESPONSE:
HTTP/1.1 200 OK
```

GET or HEAD can be accessed by Web/mobile application directly. However, PUT and DELETE are done only by Web service which is authorized by a secret key. The authorized key is sent via HTTP HEADER in each PUT or DELETE request.

6 Implementation

The central feature of our Sushi implementation is the use of Go! programming language [13]. Go! is a statically typed, compiled programming language which provides clean syntax and concurrency primitives. Moreover, it also provides rich standard libraries.

6.1 Sushi Id

Fish Glass employs the concept of content addressable storage (CAS) in which the Id of an object is a unique hash of data content, which is similar to Venti [14] project at Bell Laboratories and Centera from EMC. However, we have encapsulated additional data: dominant color of image and image pixel dimension as shown in formula 1.

$$x_j = M_i||H(j)||C(j)||D(j). \tag{1}$$

Here x_j is Sushi Id, which is generated by concatenating of M_i, $H(j)$, and $D(j)$, denoted as Fish Glass id, hash of data content, image dominant color, and dimension, respectively.

6.2 Handling Duplicate Files

Sushi follows the concept of write once and read many. Objects are never deleted or edited, which is an approach similar to Facebook's Haystack photo storage [6]. Moreover, it guarantees that no duplicate files exist in the system. The concept of CAS identifies duplicate data by hashing it. In case a Web or mobile client resends a duplicate file, Sushi will identify a duplicate image and return the existing Sushi Id of the image to the client immediately. In this case no data will be written to a disk storage. Only a pointer will be created which will point to actual physical file. Moreover, the system will track all pointers. This design of handling duplicate files not only allows Sushi to save user space resources, but also to track the most popular images.

6.3 Local File System

To ease adoption to the existing application, Fish Glass maintains POSIX file system semantic which is similar to the GIGA+ approach [2].

6.4 Security

Sushi is designed to be a loosely coupled system and can easily be deployed under Secure Socket Layer (SSL) by Hypertext Transfer Protocol Secure (HTTPS). Even though Sushi is under HTTP protocol, where URI are visible to third parties, it can stand against common snooping attacks. Thus, it is difficult to predict the dynamic name of Sushi Id.

7 Evaluation

For our evaluation purpose, we use t1.micro instances of Amazon Web Services Inc. [15]. The current spec of t1.micro node is: 1 Processor, Intel(R) Xeon(R) CPU E5430 @2.66 GHz and 8 GB storage.

7.1 Total Throughput on Single Node

Image uploads benchmarks A random image is picked up from 158,571 unique images (jpg files) for image uploads benchmarking purpose. Moreover, PUT requests are sent to upload images concurrently in 10, 50, and 100 connections on x-axis as is shown in Figs. 6, 7 and 8, respectively. Y-axis represents response time in milliseconds.
Image downloads benchmarks The way of benchmarking image downloads is different from image uploads, as shown in Figs. 9, 10, and 11, respectively. Instead of picking a random concurrency number to benchmark, we run an entire workload and see which percentage of requests is served with a particular response time. Thus, it shows the performance behavior and even availability of the Sushi under different workloads.

To prove that our design and technology choice solves the C10K problem [16], we run 10,000 concurrent requests simultaneously on t1.micro instance. The result of this benchmark is shown in Fig. 12.

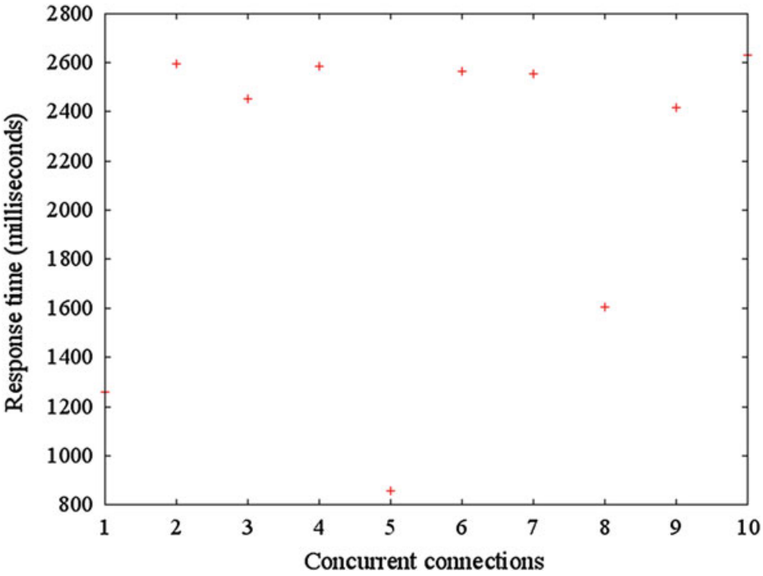


Fig. 6 10 concurrent uploads of images on single node

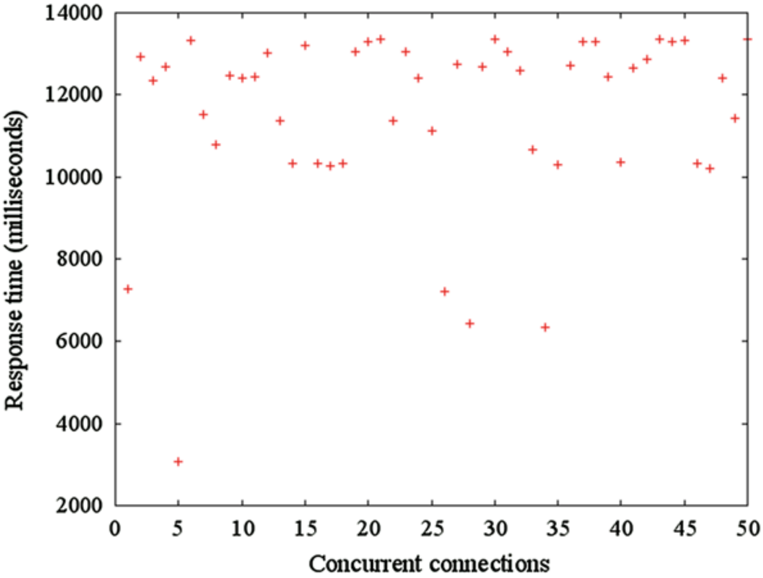


Fig. 7 50 concurrent uploads of images on single node

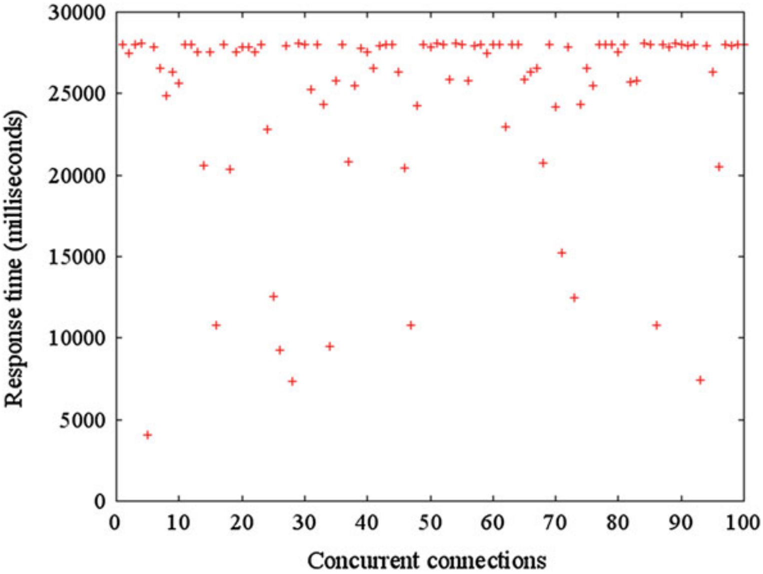


Fig. 8 100 concurrent uploads of images on single node

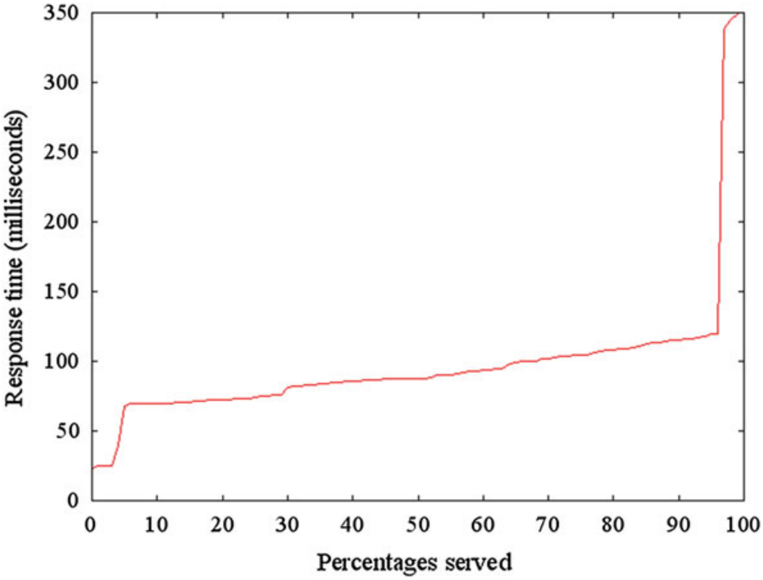


Fig. 9 100 image downloads with 100 concurrent connections on single node

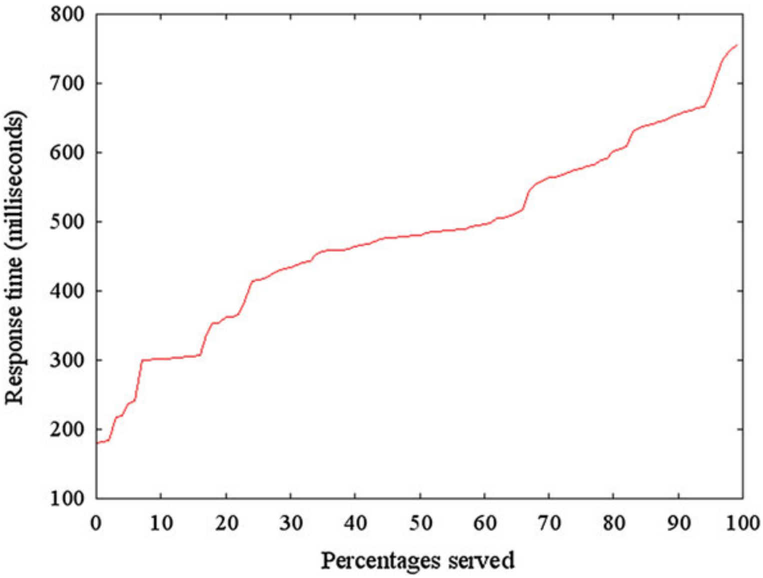


Fig. 10 500 image downloads with 500 concurrent connections on single node

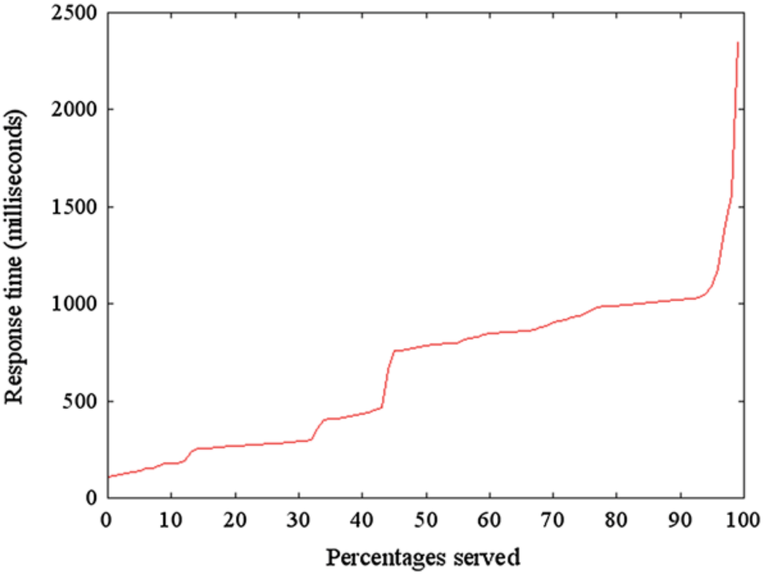


Fig. 11 1,000 image downloads with 1,000 concurrent connections on single node

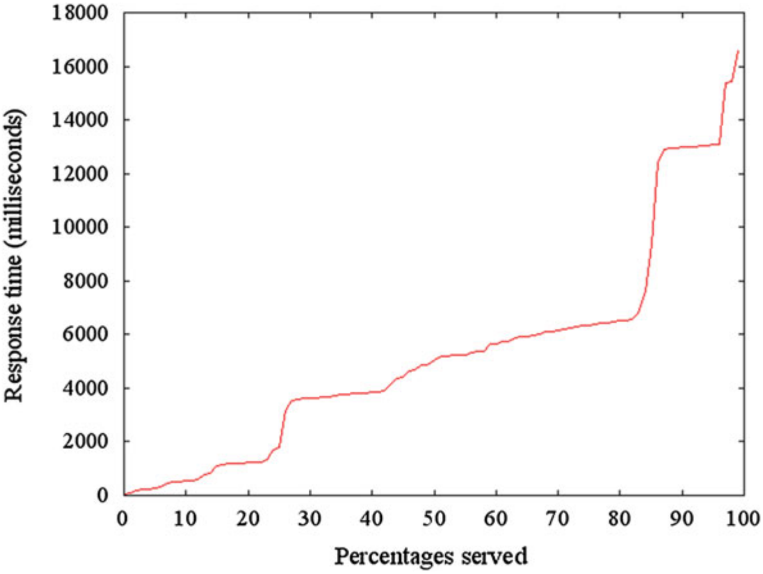


Fig. 12 10,000 image downloads with 10,000 concurrent connections on single node

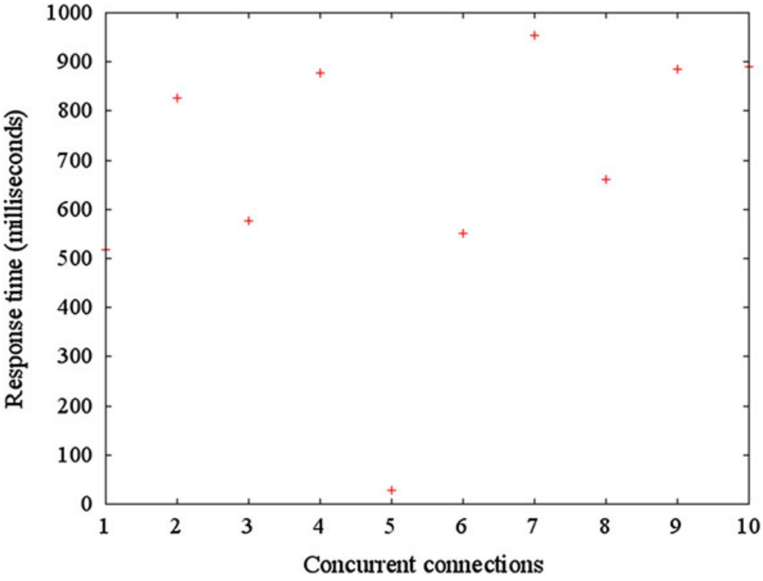


Fig. 13 10 concurrent uploads of images on cluster

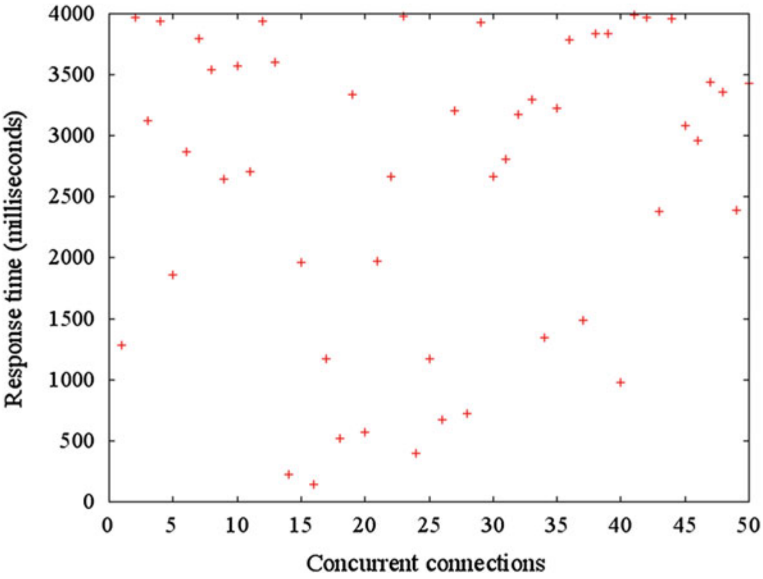


Fig. 14 50 concurrent uploads of images on cluster

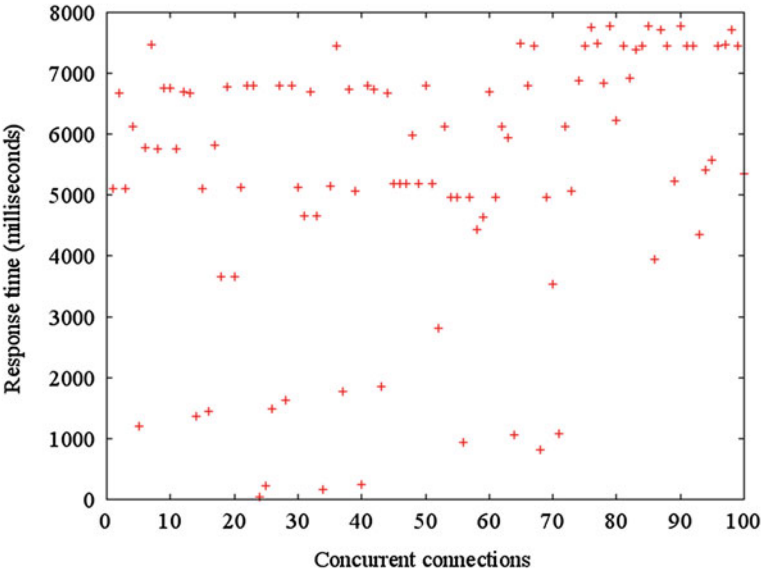


Fig. 15 100 concurrent uploads of images on cluster

7.2 Total Throughput on Cluster

Image uploads benchmarks In this benchmark we use only two-nodes cluster setup. However, in the Sushi, we can scale each component into each node or cluster nodes as shown in Fig. 4. That is how we achieve scalability and performance even in commodity servers.

Image upload benchmarks on cluster setup are shown in Figs. 13, 14 and 15.

If we compare image uploads on single node versus cluster, the differences are 2.8 times in 10, 3.5 times in 50, and 3.75 times in 100 concurrent connections, respectively.

8 Conclusion

In order to get a good balance between complexity and performance of image-surfing in Web and mobile applications, we have presented our novel solution called Sushi. The presented Sushi design enables small to medium service providers to implement efficient image-surfing in their applications. This is because the Sushi design is nonblocking and was specifically designed for the task instead of just being a workaround like caching-based approaches. Sushi's GO!-based implementation allows it to utilize the most out of multicore while using its nonblocking I/O. The implementation allows Sushi to scale in tens of thousands of concurrent connections making it ideal for image-surfing in Web and mobile services.

References

1. Internet World Stats. http://www.internetworldstats.com/stats.htm
2. Patil, S.V., Gibson, G.A., Lang, S., Polte, M.: GIGA+: scalable directories for shared file systems. In: PDSW '07 Proceedings of the 2nd International Workshop on Petascale Data Storage: Held in Conjunction with Supercomputing'07, pp. 26–29. ACM, New York (2007)
3. Graziano, P.: Speed up your web site with varnish. Linux J. **2013**, 4 (2013)
4. Amarakeerthi, S., Liyanage, K., Cohen, M.: Delay pools-based uninterrupted and dynamic bandwidth distribution of Squid proxy cache. In: HCCE '12 Proceedings of the 2012 Joint International Conference on Human-Centered Computer Environments. pp. 244–246. ACM, New York (2012)
5. Chi, X., Liu, B., Niu, Q., Wu, Q.: Web load balance and cache optimization design based Nginx under high-concurrency environment. In: ICDMA, pp. 1029–1032 (2012)
6. Beaver, D., Sanjeev, K., Beaver, D., Kumar, S., Li, H.C., Sobel, J., Vajgel, P.: Finding a needle in haystack: facebook's photo storage. OSDI. **10**, 1–8 (2010)
7. Eastlake, 3rd.D., Jones, P.: US secure hash algorithm 1 (SHA1). RFC Editor, United States (2001)
8. Ongaro, D., Ousterhout, J.: In search of an understandable consensus algorithm. In: Proceedings of the ATC14, USENIX Annual Technical Conference (2014)
9. Lakshman, A., Malik, P.: Cassandra: a decentralized structured storage system. SIGOPS Oper. Syst. Rev. **44**, 35–40 (2010)

10. DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., Vogels, W.: Dynamo: amazons highly available key-value store. In: SOSP '07 Proceedings of Twenty-First ACM SIGOPS Symposium on Operating Systems Principles, pp. 205–220. ACM, New York (2007)
11. Rosenblum, M., Ousterhout, J.K.: The design and implementation of a log-structured file system. ACM Trans. Comput. Syst. **10**, 26–52 (1992)
12. Chen, S., Nepal, S., Bouguettaya, A.: A smart user interface for service-oriented web. Lect. Notes Comput. Sci. **6724**, 318–330 (2011)
13. Clark, K.L., McCabe, F.G.: Go! for multi-threaded deliberative agents. Lect. Notes Comput. Sci. **2990**, 54–75 (2004)
14. Quinlan, S., DorwardVenti, S.: A new approach to archival data storage. In: FAST '02 Proceedings of the 1st USENIX Conference on File and Storage Technologies, Article No. 7. USENIX Association Berkeley, CA (2002)
15. Amazon Web Services Inc. http://aws.amazon.com/
16. The C10K problem. http://www.kegel.com/c10k.html

Verification of Identity Based on Palm Vein and Palm-Print

Mariusz Kubanek and Dorota Smorawa

Abstract This paper presents a biometric system implementing identity verification based on palm vein and palm-print. The paper includes suggestions of our own algorithms as well as palm feature extraction methods and their specific coding for verification. The obtained results show that the human palm carries a lot of useful information and, appropriately modified, can become an interesting proposal among biometric systems, also carrying out the verification process with a large number of users.

Keywords Palm-print · Palm vein · User verification · User identification

1 Introduction

Images are a special form of data and their specification has not been fully explored. However, digital image processing has been gaining more popularity in recent years. Vision systems appear in everyday life, starting from monitoring systems to biometric passports. Thanks to the use of appropriate image processing operations, information which is usually not visible to a human visual system can be obtained.

People identification is an important problem in many computer systems and electronic systems. The well-known methods of identification, such as entering a PIN number, entering login and password, or using ID cards have many difficulties

M. Kubanek (✉) · D. Smorawa
Institute of Computer and Information Science, Czestochowa University of Technology,
Dabrowskiego Street 73, 42-200 Czestochowa, Poland
e-mail: mariusz.kubanek@icis.pcz.pl

D. Smorawa
e-mail: dorota.smorawa@icis.pcz.pl

M. Kubanek
Department of Computer Science, European University of Information Technology
and Economisc in Warsaw, Bialostocka Street 22, 03-741 Warsaw, Poland
http://icis.pcz.pl/~mkubanek/

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_12

and disadvantages. It is easy to forget PIN numbers, passwords, as well as lose identification cards. In addition, the card can be stolen and protecting passwords broken. Therefore, the traditional methods of identifying people are becoming less popular. On the other hand, biometric methods are gaining vast popularity in identification and verification of people [1]. These methods use the digital measurement of certain physical and behavioral characteristics of humans and compare them with the pattern stored in the database. Until now, many biometric methods which enable identification and verification of people have been developed [1]. From among all of these methods, palm vein [2–5] and palm-prints [6, 7] recognition deserves special attention.

In paper [8] the author shows the way of identification which is based only on basis metacarpalis. A person is identified on the basis of a palm part which is located between the wrist and the fingers. The image of the palm, as in most works of this type, is subjected to a segmentation, which is the extraction of palm-prints. In the following work over the image, the author uses Gabor filters to obtain a more accurate frame of the palm-prints. The final step is to compare the image in the database with the received image of the palm. Studies were conducted on about 200 people. There were taken 20 pictures of the left and 20 pictures of the right palm of each person participating in the studies. A rejection rate of entitled person was only FRR $= 3\,\%$, and FAR rate $= 0.1\,\%$.

We propose a method of verification of identity, which in addition to palm-prints also includes palm vein pattern. This method involves nontypical features like the shape of the main lines, shown on the inner side of the palm and the vein pattern. Information about the features of a palm is calculated on the basis of the palm picture. The standard device for image acquisition of a palm consists of a closed-circuit television camera, an infrared floodlight, and a special plate which is equipped with seven rings for palm positioning. This work is a continuation of the studies described in the work [9].

2 Image Acquisition

Prior to building the system that performs identity verification based on features provided by palm picture it is necessary to determine in what conditions and in what environment the system will work. For the analysis of palm image may be used a specially prepared plate with pins placed on the surface of it [9]. The pins should provide the same arrangements on the plate for each of the analyzed palms, which at a later stage would greatly facilitate the process of measuring features of a palm. In addition, the camera is located at a constant height and constant position, which simplifies the process of preprocessing the image, because each image is made from the same distance and it is not necessary to scale the image. Unfortunately, the position of the palm is not comfortable for the user, but at this stage of research, the use of tiles is required. Further studies will be conducted on a more comfortable scanners. Figure 1 shows our test stand.

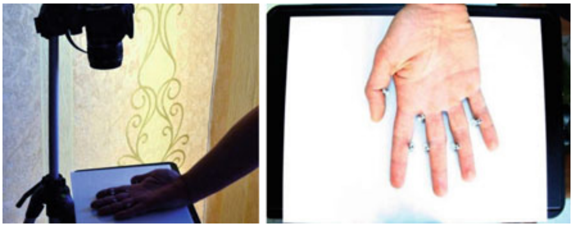


Fig. 1 The test stand

The obtained palm image must undergo the indispensable pretreatment processes in order to adjust the image to the requirements of the measurement procedure of geometric features.

3 Detection of Main Lines of the Palm

Searching for simplicity in proceedings, and thus accepting only the main palm-prints analysis, an interesting solution could be the use of a simple directional edge detector. With the appropriate settings of detector coefficients it is possible to omit little, short, faint lines. However, there is a problem because such a clearly visible palm line will have designated edges on both its sides, which does not facilitate further work. However, at this time we can use a morphological operation—dilation, which will connect the separate edge lines.

Dilation greatly increases the thickness of lines, but the shape and location of major palm-prints are not changed. In order to interchangeably define the palm-prints shapes, these bold elements should be brought closer by one-pixel thick lines. This can be done by using the linear Hough transform. The transform allows the detection of the straight lines in binary image. With appropriate parameters (the minimum length of searched palm-prints should be adjusted) the output effect of Hough transform will be the whole family of lines gathered around the bold palm-prints. Taking into account the distribution of nodes in the Hough space, we select

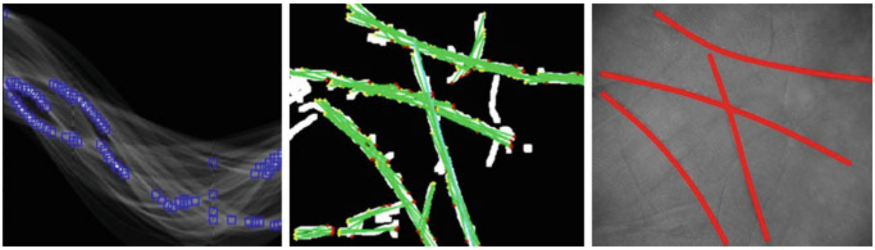


Fig. 2 Designated space of Hough transform on the *left*, applied operation of lines detection on the basis of the transform in the *middle* and approximation operation on the *right*

only those lines whose length is greater than the preset threshold. Points defining half the distance between the beginning and end of each of the selected lines and the coordinates of beginnings and ends of lines are to approximate the shape of the base line [9]. Figure 2 illustrates the effect of designating major palm-prints with the usage of the linear Hough transform.

4 Coding of Appointed Features

Coding of palm-prints of the palm can be executed starting from dividing the input image into a fixed number of square sub-images. In this work the size of image consisting of palm-prints is 60 × 60 mm. At these values the input image can be divided into 144 sub-images.

Each sub-image has one observation symbol. Coding of palm-prints consists in the choice of only these sub-images which consist of any of fixed palm-prints. Forming the observation vector the sub-images are chosen according to the approved numeration. If a given sub-image contains a line, then besides the number of sub-images, in the observation vector the information about the average angle where a given piece of a line is arranged is relative to the horizontal axis. In order to limit the observation symbols connected with coding the angles, defined between palm-prints and the horizontal axis, the division of semi-circle (180°) is accepted to a fixed number of slices (e.g., 16). In such a way for every angle one of 16 observation values is assigned. The scheme of such coding is shown in Fig. 3.

Based on the above-mentioned scheme, the sample observation vector begins from the third value for which the angle code is 13, the next observations are 4th and 16th values with the angle code 13, and in consequence the next is 17th with the angle code 14, etc. If in the given sub-image there are two or more lines, then besides the number of observed sub-areas, the angle codes of all lines are assigned (starting from the lowest lines). The resultant angle is assigned on the basis of a tangent to a curve (describing the palm-print) in points P1 (the beginning of the line), P2 (the middle of the line), and P3 (the end of the line) within a given sub-image. In the next steps the average angle is determined from the angles $\alpha 1$, $\alpha 2$ and $\alpha 3$.

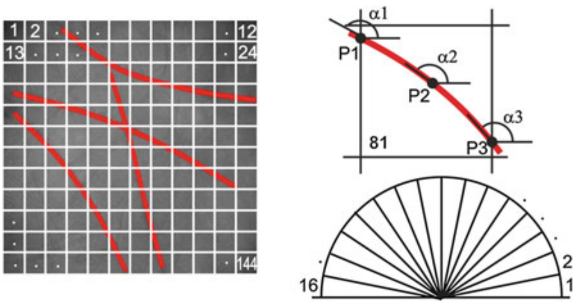


Fig. 3 The scheme of procedure during assigning the observation vectors using this coding method

The presented way of coding assumes the possibility of erroneous detection of main palm-prints, which determines an incomplete line or its entire omission. The essential element in such a way of coding is the acceptable variable number of observations, which in case of vector measurement using the distance method, would introduce the complement of vector size (e.g., unities, zeros), causing too much similitude of data. Using the teaching-verifying tool in the case of Hidden Markov Models [10–12], the possibility of generalizing the patterns is kept, despite rapidly changing observation vectors of certain characteristics of the image.

5 Palm Vein Pattern Detection

During image acquisition of palm vein, the wavelength of light in the near infrared is used. Using the camera with a matrix for near-infrared and infrared radiation with a range of about 850 nm allows to see invisible attributes in visible light. Contained hemoglobin in the blood absorbs near-infrared light, giving a natural contrast. In Fig. 4 the image of a palm seen in the infrared are shown.

After the acquisition of images, the first step is to improve the contrast. This is an important step because it significantly increases the visibility of veins. Various algorithms to improve the contrast of course give different results, as shown in Fig. 5. We should choose an appropriate algorithm to improve the contrast in order to further the proper functioning of our method.

The next step is to make the edge detection. This can be done using, for example, EDGE function. Of course, we assume the same settings for all users. To detect pattern veins we use the same algorithm that was used to detect the main lines of the palm. We use designated Hough transform coefficients as a feature vector. The designated position of the veins is shown in Fig. 6.

On the basis of biometric features provided by the palm vein pattern, we perform the verification and identification of identity. Recognition consists in selection of the feature vector most similar to the input vector.

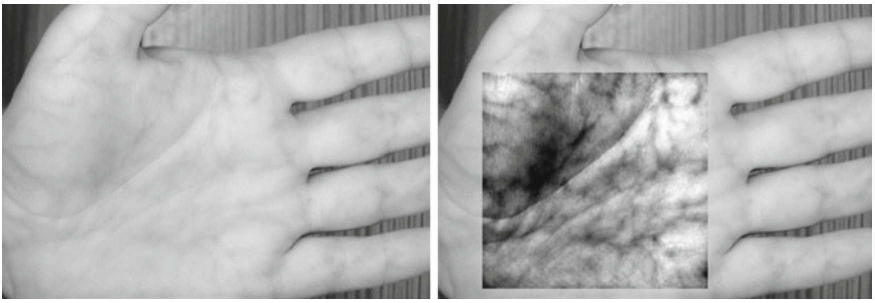


Fig. 4 Image of a palm seen in the infrared

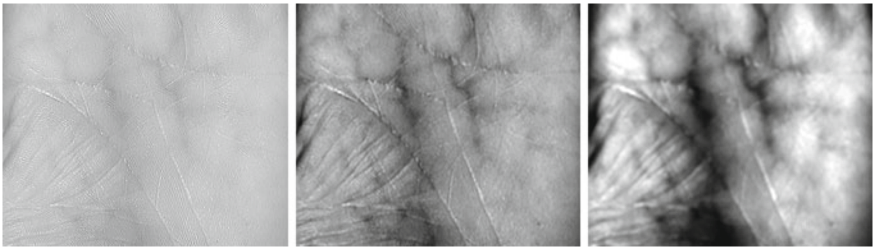


Fig. 5 Different results of contrast enhancement

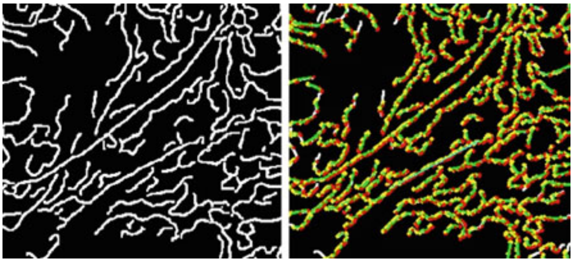


Fig. 6 Designated position of the veins

6 Experimental Results

The research to determine the effectiveness of the identity verification on the basis of the palm veins and main palm-prints are conducted for the usefulness of these nontypical features also to biometric systems.

The first of the research connected with the human palms is to determine the correctness of the identity verification on the basis of only main lines of palm. The study was to determine the false rejection rate and the false acceptance rate for two different databases: Casia Multi-Spectral Palmprint Database and our database. 100 different users were tested for whom 6 photos were downloaded (three for left hand and three for right hand). All pictures that form input data were taken in the same conditions and using the same equipment. The results of errors are in Table 1.

The second of the research connected with the human palms is to determine the correctness of the identity verification on the basis of the main lines of the palm and vein patterns of the palm. The study also was to determine the false rejection rate and the false acceptance rate for two different databases: Casia Multi-Spectral Palmprint Database and our database. Because we did not have a larger number of images for

Table 1 Results of tests for identity verification based on main lines of the palm

Database	Left hand		Right hand	
	FAR (%)	FRR (%)	FAR (%)	FRR (%)
Our database	9.75	13.25	8.75	12.50
CASIA MSPD	10.50	14.25	9.50	14.50

Table 2 Results of tests for identity verification based on main lines of the palm and vein patterns of the palm

Database	Left hand		Right hand	
	FAR (%)	FRR (%)	FAR (%)	FRR (%)
Our database	0.75	1.5	0.50	1.25
CASIA MSPD	1.00	1.75	0.75	1.50

palms, made in near infrared, only 50 different users were tested for whom 6 photos were downloaded (three for left hand and three for right hand). It should be noted that correct verification occurs only if both vectors (vector of coded main lines of the palm and vector of coded palm vein) come from the same user. The results of errors are in Table 2.

7 Conclusion and Future Work

In this paper, the biometric system for identity verification is shown on the basis of palm vein patterns and main palm-prints. Several solutions are proposed connected with the analysis of the image for extraction of searching features and the way of connecting assigned main lines and vein pattern features of a palm.

Conducted studies show that the analysis of features of main lines of palm does not give enough safety. But if you take into consideration the features given by the picture of the palm vein, then by proper coding it is possible to gain better results.

If we base on palm vein image, we can expect very good results. Although we tested only 50 users, our method showed no errors for both our database and the CASIA database. Isolated cases have been associated with images of poor quality.If our assumptions are confirmed, the vein patterns can be as unique as a fingerprint pattern.

Further studies will be connected by increasing the effectiveness of the biometric system through the expansion of the number of variable characteristics to geometric features provided by the visual features coming from the detailed palm-prints and palm veins. In addition, the use of special plates will not be required during the acquisition, through the use of automatic positioning and scanning. We also plan to develop a method to encode the main lines of the palm and palm vein using a single combined vector.

Acknowledgments Portions of the research in this paper use the CASIA-MS-PalmprintV1 collected by the Chinese Academy of Sciences' Institute of Automation (CASIA) [13].

References

1. Kubanek, M., Smorawa, D., Kurkowski, M.: Using facial asymmetry properties and hidden Markov models for biometric authentication in security systems. Lect. Notes Artif. Intell., Part II **8468**, 627–638 (2014)
2. Sarkar, I., Alisherov, F., Kim, T., Bhattacharyya, D.: Palm vein authentication system: a review. Int. J. Control Autom. **3**(1), 27–34 (2010)

3. Deepamalar, M., Madheswaran, M.: An enhanced Palm vein recognition system using multilevel fusion of multimodal features and adaptive resonance theory. Int. J. Comput. Appl. **1**(20), 95–101 (2010)
4. Zhang, H., Hu, D.: A Palm vein recognition system. In: 2010 International Conference on Intelligent Computation Technology and Automation (ICICTA), vol. 1, pp. 285–288 (2010)
5. Soliman, H., Mohamed, A.S., Atwan, A.: Feature level fusion of Palm veins and signature biometrics. Int. J. Video Image Process. Netw. Secur. IJVIPNS-IJENS **12**(1), 28–39 (2012)
6. Choras, M., Kozik, R.: Contactless palmprint and knuckle biometrics for mobile devices. Pattern Anal. Appl. **15**(1), 73–85 (2012)
7. Choras, M., Kozik, R.: Fast feature extractors for Palmprint biometrics. In: Chaki, N., et al. (eds.) Computer Information Systems—Analysis and Technologies, Communications in Computer and Information Science CCIS, pp. 121–127. Springer (2011)
8. Fatric, I., Ribaric, S.: Colour-based palmprint verification—an experiment. In: Watteyne, T., Yazidi, A. (eds.) Proceedings of IEEE 14th Mediterranean Electrotechnical Conference, Ajaccio, pp. 890–895 (2008)
9. Kubanek, M., Smorawa, D., Adrjanowicz, L.: Users verification based on palm-prints and hand geometry with hidden Markov models. Lect. Notes Artif. Intell. **7895**, 275–285 (2013)
10. Bobulski, J.: Selection of parameters of HMM. Meas., Autom., Control **10**, 844–846 (2009)
11. Bobulski, J., Adrjanowicz, L.: Two-dimensional hidden Markov models for pattern recognition. Lect. Notes Artif. Intell., Part I, **7894**, 515–523 (2013)
12. Kubanek, M., Bobulski, J., Adrjanowicz, L.: Characteristics of the use of coupled hidden Markov models for audio-visual Polish speech recognition. Bull. Pol. Acad. Sci.—Tech. Sci. **60**(2), 307–316 (2012)
13. http://biometrics.idealtest.org

The Concept of Automation in Conventional Systems Creation Applied to the Preliminary Aircraft Design

Nikolay Borgest, Maksim Korovin, Andrey Gromov and Aleksey Gromov

Abstract The paper describes the concept of an approach to the automation of typical design decisions in the preliminary design of an aircraft applied to the design of a conventional regional multipurpose aircraft. The description of a CAD master model as a basis for finite element structure optimization is given. The issues regarding parameterization and optimization of the model in the process of refining the design project are stated. The relations between the subject of design process and the CAD metamodel based on CATIA are examined in application to the automation of designing. An interaction schema for the computational, geometrical, and heuristic components of the designing decision support system on the basis of ontological domain description is given.

Keywords Automation · Decision making · CAD · Ontology · Aircraft · Design

1 Introduction

Major changes in commercial airplane development have occurred worldwide during the several past decades, which place increased emphasis on product quality and cost. The problem of accurate estimation of the most essential aircraft characteristics on the preliminary design stage can be addressed by the reorganization of the design process to utilize finite element experiments and formal mathematic optimization into the very first stages of the design process [1].

N. Borgest (✉) · M. Korovin · A. Gromov · A. Gromov
Samara State Aerospace University, Samara, Russia
e-mail: borgest@yandex.ru

M. Korovin
e-mail: maks.korovin@gmail.com

A. Gromov
e-mail: gomer191@gmail.com

A. Gromov
e-mail: gomer191@mail.ru

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_13

The initial aircraft configuration is defined and parametrically optimized during the conceptual design stage. A set of design variables typically size the major components of the aircraft configuration. However, the initial set of design variables does not completely map into a 3D computer-aided design (CAD) representation, which can be used for further estimation of the design variables, including finite element analysis [2].

Advancing high-fidelity geometry into the early stages of the design process has the potential to mitigate the amount of mistakes and inaccuracies typically made due to the low model fidelity [3].

The scope of application of the method below includes a limited number of airframe parts such as wing, fuselage, and empennage, because design features of landing gear and power plant are determined by the commercial requirements, operation conditions, and other specific factors beyond the scope of aircraft structure analysis.

2 Scope

Modern CAD systems are gradually making a transition from being a tool toward serving as an “intelligent partner” for the designer. High level of formalization of the common design procedures allows CAD systems to perform the ever-increasing amount of tasks without the need of the human designer’s involvement. CAD knowledge is an essential quality for a modern engineer. However, modern heavy CAD systems have more and more complex interfaces that make the task of familiarization quite challenging. Even for the engineers, familiar with the CAD interface within their daily tasks range, the task of identifying the right instruments for an unfamiliar task may present a challenge.

One of the methods of simplifying complex CAD interfaces might be an addition of a voice control mechanism on top of the classic user interface. In this paper, we will examine the implication of such mechanism in the field of preliminary aircraft design.

During the last years, the idea of the voice-enabled CAD control had been quite popular and several implications have been reported in various manuscripts, such as [4–6]. The common problem in the field of aircraft designing is the presence of ambiguity regarding terms and definitions in the field, further complicated by the heavy use of professional slang. To overcome this problem, an implementation of an additional slang translation module could be implemented as shown on Fig. 1 [7].

Provided that the process of designing is fully formalized within a certain set of tasks, it is possible to create a system that will be able to perform parametric optimization (including structural) of a predefined parameterized model. During the optimization of design parameters, an instance of the parameterized metamodel, that potentially includes all the possible solutions, is generated based on the input data to satisfy the design conditions.

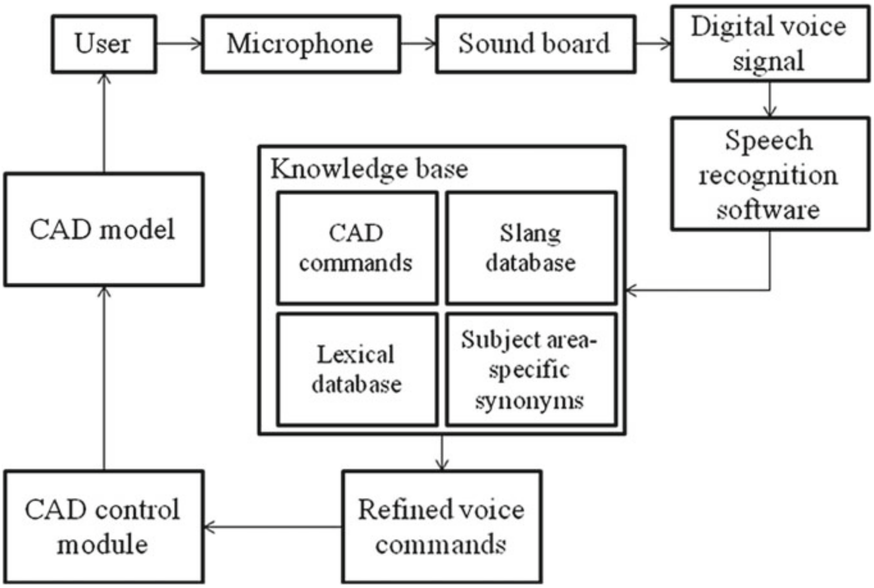


Fig. 1 Model of the voice-enabled CAD control

It is well known that designing is a field, where a lot of heuristic solutions are implemented. However, in the field of traditional structures designing most of the initially heuristic approaches have been formalized, which makes it possible for an automated system to use them successfully.

One of the main problems with voice-enabled CAD control is the amount of controlled parameters needed to produce a 3D model. This issue can be addressed by implementation of highly parameterized models that allow mapping of a relatively small set of design variables onto a full 3D representation of an airplane part, or high level primitives.

3 Approach

3.1 Method

The question of flexibility and simplicity balance in engineering design tools had always been an issue. On the one hand, the use of high level CAD primitives that can be adapted to a variety of tasks looks tempting, but on the other hand, the designer must bear in mind that highly parameterized complex models with inbuilt logic are more expensive and take more time to develop in comparison to trivial CAD models. The nature of engineering dictates the necessity for periodic revisions of the model. Furthermore, due to the presence of a certain ambiguity in terms and definition between different branches of the engineering domain, the task of creation of a multidisciplinary model that is maintainable by any domain expert, not just its creator becomes even more complex. We propose an ontology-aided solution for the complex parameterized CAD model logic description based on the thesaurus of the knowledge domain that is currently being applied to the preliminary aircraft design decision support system.

The approach toward realization of the method is disclosed in Sect. 4.

3.2 Tools

The ever-growing need for the time-effort efficiency increase in designing requires new tools that will allow engineers to work with CAD and, if necessary, CAE primitives on a higher abstraction level, or high level primitives (HLP) [8]. The use of libraries, containing HLPs, like wings, various fuselage parts, empennages, and so on, with inbuilt engineering decision support systems, will allow engineers to interact with the CAD/CAE system with only the predefined design and analysis parameters thus eliminating the need to interact on the lower level of abstraction.

The principles of the HLP-driven design are presented in a number of manuscripts, such as La Rocca [9], Liersch [10], Ledermann [11], and Lin [12].

One of the most common fields for the specific CAD instruments implication, both for commercial CAD systems and standalone tools, is the aircraft design. The use of knowledge-based engineering (KBE) support tools in the modern CAD, that are frequently used in the aircraft designing process, such as CATIA from Dassault systems, Pro Engineer from PTX, AutoCAD from Autodesk, or NX from Siemens allows the creation of highly parameterized models with complex internal logics that are further to be used as HLP in the designing process. However, modern CAD systems do not have descriptive abilities to support complex models with metadata, sufficient for the automated assortment with a view to further reuse of such kind of models. In the field of aircraft design, the selection of a proper HPL can be further complicated by the fact that visually similar models may have substantial differences in structure and/or process requirements, as demonstrated on Fig. 2.

Here, two seemingly similar models have different inner structures as a consequence of different technological and aerodynamic requirements. Thus, to perform a successful selection of a suitable HLP for the further optimization within the design process, an implementation of HLP metadata is needed. We define major HLP properties, affecting their possible applications, with statements, such as “Wing of a low speed regional aircraft”, “Wing structure—single-spar semimonocoque,” and so on. The terms, comprising the statements, are implemented in the airplane thesaurus to ensure consistency of terminology and to resolve any ambiguity in descriptions [13].

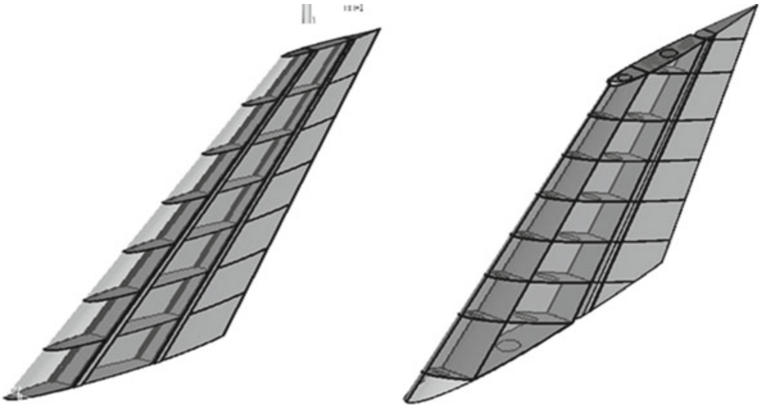


Fig. 2 Models of the vertical tail units for high and low speed airplanes

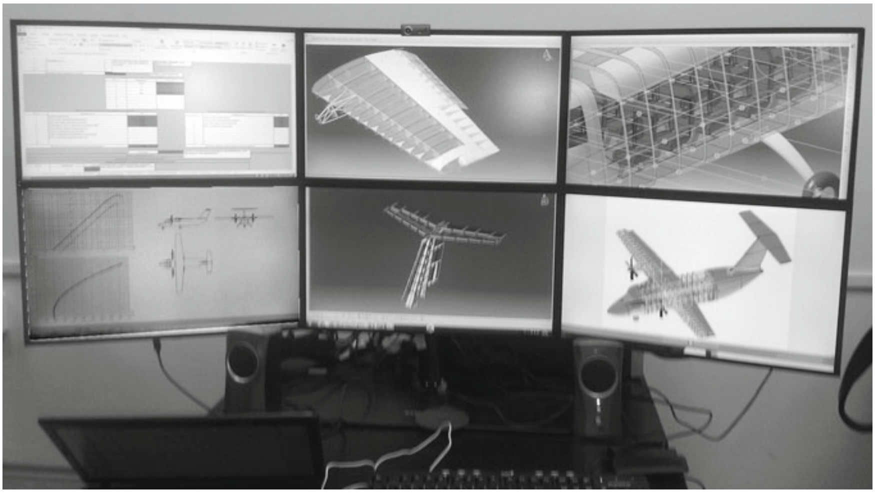


Fig. 3 Hardware multiscreen interface of the system

The base design variables set, defining the HLP state, is dependent on the intended model usage, described in the metadata, so it is possible to combine the metadata and the design variables in one external database. The view of the systems interface, implemented on a computer with six monitor setup is presented on Fig. 3.

The information about the design variables, necessary to create a high-fidelity CAD model is stored externally. The separation between the model and the metadata allows for a natural translation toward the cloud CAD usage. Having a complete within a problem to be solved formal description on all stages of the products life cycle allows natural advanced CAD integration into PLM systems of organizations.

4 Master Model Definition

4.1 Master Model Overview

The model, developed in CATIA, using the inbuilt system capabilities is shown on the Fig. 4. The model is based on the technical specification designed by the scientific-production association "Aero Volga".

Changing the input parameters, such as the cruising speed and altitude initiates the automated regeneration of the model to comply with the changed requirements based on the inbuilt reasoning (Fig. 5).

The macroprimitives used in the model are:

- Wing (which includes the external geometry and the primary structure);
- Fuselage (with external geometry, primary structure, and internal layouts for different airplane configurations, as well as the control surfaces for aerodynamic calculations in different flight conditions);
- Empennage (with separate vertical and horizontal tail units);
- Engine models (at current stage for the aerodynamic calculations only).

The model is built using absolute system of coordinates. The initiation of the addition of a macroprimitive triggers its drawing. After it is drawn, it is positioned based on the conjugations between the parts. All the internal elements of the primary structure, such as ribs, frames, stringers, or spars are generated based on the external geometry rather than calculated. This is necessary to avoid unintended surface intersections, that could complicate the finite element mesh generation.

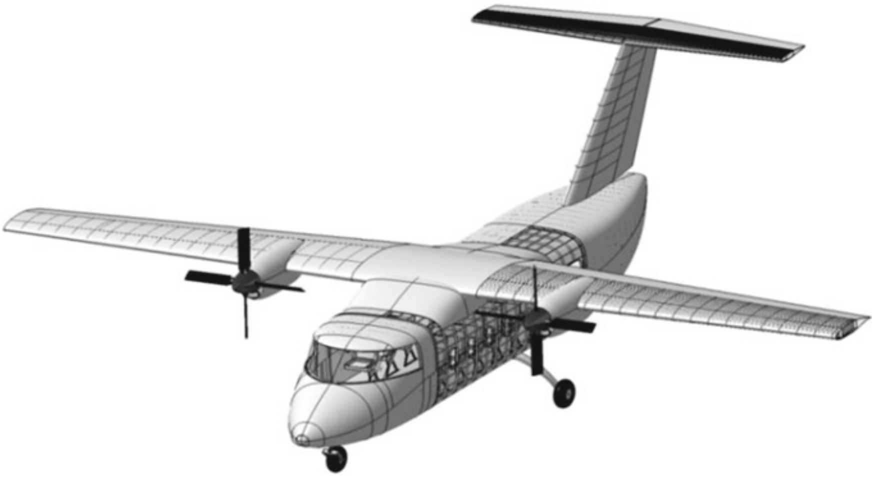


Fig. 4 Automatically generated CAD model of a regional airplane

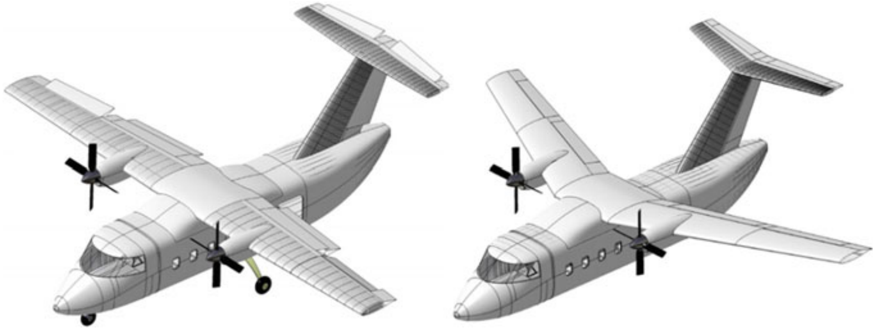


Fig. 5 Instances of the master model for different inputs

All the input design parameters are stored in the external database, currently in an MS Excel file. The interaction between the user and the model is done through the interface, implemented in Visual Basic. All the inputs from the user change the state of the model that triggers the regeneration of the model. The use of external data storage allows implementation of a large amount of reference data into the model, as well as some design decisions and methods.

4.2 Generation of HLPs

The design model for the half-wing master geometry of a regional aircraft is shown on Fig. 6.

Due to the complexity of the model it is necessary to provide end users with tools that will enable the alteration of the master model beyond the input parameters alteration. This task is additionally complicated by the fact that even if the model itself is coded (inside CATIA KBE module) with names of specific parts like "wing span," "fuselage frame spacing," etc., there is still a lot of ambiguity regarding less commonly used terms and definitions. To resolve this issue, an implementation of a thesaurus for the HLP as a component for the aircraft thesaurus is proposed.

The generated macroprimitive instance may be used in numerical experiments to refine the initial conditions of the model through optimization [14]. The final result of the macroprimitive instance generation is shown on Fig. 4. The generated HLP can further be optimized based on the finite element analysis (FEA). However, successful implementation of FEA does usually require a high level expertize from the engineer [15]. The task of creating a FEA model is still mostly based on the user's experience [16]. The finite element mesh definition is one of the most complex and time-consuming stages of FEA [17]. The problem of defining the mesh element

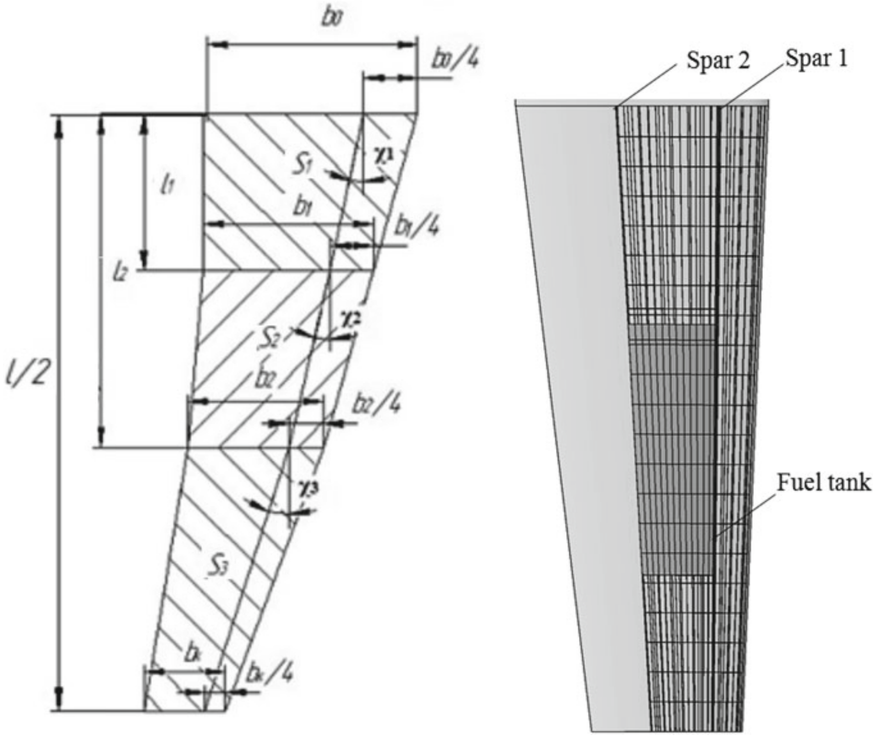


Fig. 6 Aircraft wing macroprimitive structure and generated instance

types and properties can be at least partially addressed by addition of suggested element types and properties as well as some reference data into the geometry models metadata within the HLPs [18].

An example of the finite element analysis of the wing is demonstrated on Fig. 7.

4.3 Optimization of an HLP

Analysis of the basic requirements for the airplane structure shows that they have a critical impact on the airplane weight, production qualities, and usability. The aerodynamic and the body-mass arrangement demands should be addressed as constraints that should be satisfied. According to the basic principle of airplane designing, presented in the form of the airplane existence equation, the enhancement of any quality of an aircraft demands addition of extra material and therefore affects its weight.

The method for the HLP optimization is based on the variable density model approach that was described in [19]. A model body consists of elements with variable density. The volume density defines the modulus of elasticity and tensile strength.

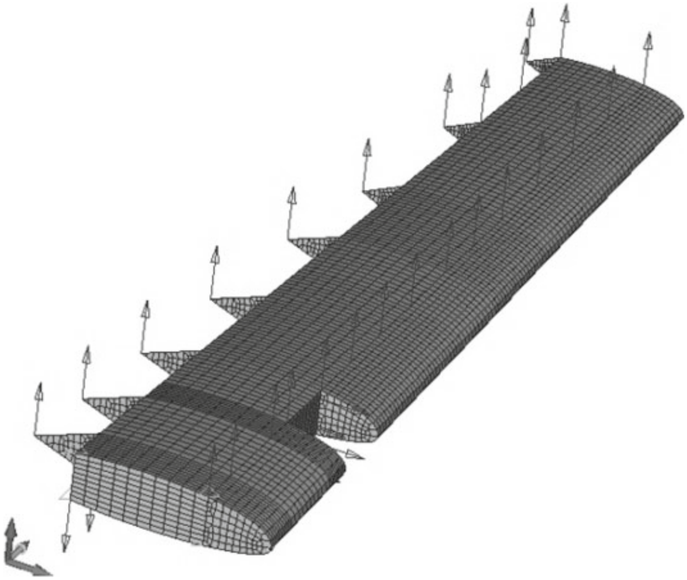


Fig. 7 Finite element analysis of a left wing panel

Optimization of the density distribution within the model gives the theoretically optimal structure that can be further processed to comply with the technological requirements.

5 Conclusion

The application of an ontological model as a basis for the generation of a 3D CAD model based on a set of macroprimitives has been shown. The description of HLPs is composed on the basis of thesaurus of the aircraft design domain. Thesaurus is used to attribute models parameters and simultaneously serves as the prime source of terms and definitions for the HLP generation interface and a means of linking of terms and parameters from databases, containing information about past projects. The addition of an external description can drastically increase the maintenance capabilities of the macroprimitives based on the fact that the presence of accurate semantically correct descriptions allow domain experts to interact with the macroprimitive internal logic and structure without the necessity to consult with the primitive creator [20]. Further use of the macroprimitive ontological model may include an addition of specific output conditions which would allow the use of an intellectual interface, capable of solving the most common design tasks without the need of human assistance.

The work has been carried out in Samara State Aerospace University with support from Ministry of Education and Science of Russian Federation.

References

1. Blessing, L.T.M., Chakrabarti, A.: DRM, a Design Research Methodology, 413 p. Springer, New York (2009)
2. Dattoma, V., De Giorgi, M., Giancane, S., Manco, P., Morabito, A.E.: A parametric-associative modeling of aeronautical concepts for structural optimization. Adv. Eng. Softw. **50**(1), 97–109 (2012)
3. Lazzara, D.S., Haimes, R., Willcox, K.: Multifidelity geometry and analysis in aircraft conceptual design. In: 19th AIAA Computational Fluid Dynamics 22–25 June 2009, San Antonio, Texas (2009)
4. Ames, B.B.: Voice-command CAD lets you speak in 3D. Autom. Control 1/8/2001
5. Bernsen, N., Dybkjaer, L.: Is speech the right thing for your application? In: 5th International Conference on Spoken Language Processing (1998)
6. Chu, C.-C.P., Dani, T.H., Gadh, R.: Multi-sensory user interface for a virtual-reality-based computer-aided design system. Comput.-Aided Des. **29**(10), 709–725 (1997)
7. Xue, S., Kou, X.Y.: Natural voice-enabled CAD: modeling via natural discourse. Comput. -Aided Des. Appl. **6**(1), 125–136 (2009)
8. Tarkian, M.: Design automation for multidisciplinary optimization—a high level CAD template approach. Linköping Studies in Science and Technology. Dissertations, Linköping University. No. 1479, 116 p. (2012)
9. Guarino, N.: Formal ontology and information systems. In: Proceedings of FOIS'98, Trento, Italy, 6–8 June 1998, pp. 3–15. IOS Press, Amsterdam (1998)
10. La Rocca, G., Van Tooren, M.: Knowledge-based engineering approach to support aircraft multidisciplinary design and optimization. J. Aircr. **46**(6), 1875–1885 (2009)
11. Liersch, C., Hepperle, M.: A unified approach for multidisciplinary preliminary aircraft design. In: CEAS European Air and Space Conference, 26–29 October 2009, Manchester (2010)
12. Ledermann, C., Hanske, C., Wenzel, J., Ermanni, P., Kelm, R.: Associative parametric CAE methods in the aircraft pre-design. Aerosp. Sci. Technol. **9**(7), 641–651 (2005)
13. McLauchlan, M.: Thesauruses for prepositional phrase attachment. In: Proceedings of CoNLL-2004, pp. 73–80. Boston
14. George, M.T., Steven, J.F.: Knowledge-based assistance for finite-element modeling. IEEE Intell. Syst. **11**(3), 23–32 (1996)
15. Wriggers, P., Siplivaya, M., Joukova, I., Slivin, R.: Intelligent support of engineering analysis using ontology and case-based reasoning. Eng. Appl. Artif. Intell. **20**, 709–720 (2008)
16. Pinfold, M., Chapman, C.: The application of KBE techniques to the FE model creation of an automotive body structure. Comput. Ind. **44**(1), 1–10 (2001)
17. Bojan, D.: Finite element mesh design expert system. Knowl.-Based Syst. **15**, 315–322 (2002)
18. S, Wei, et al.: A Framework for automated finite element analysis with an ontology-based approach. J. Mech. Sci. Technol. **23**, 3209–3220 (2009)
19. Komarov, V.A.: Concurrent design. Ontology of designing. No. 3(5), pp. 8–23 (2012) (In Russian)
20. Borgest, N.M. and others: Robot-designer: fantasy and reality. Ontology of designing. No. 4 (6), pp. 73–94 (2012) (In Russian)

Video Stream Analysis for Fish Detection and Classification

Paweł Forczmański, Adam Nowosielski and Paweł Marczeski

Abstract The paper presents a concept of automatic video stream analysis which leads to the detection and tracking of specific objects, namely fish silhouettes that move in a water tank. It is one of the most important problems to be taken into consideration during the environmental studies. The paper includes mathematical principles related to adaptive background model and object classifier. The approach involves Gaussian Mixture Model for background elimination and foreground objects extraction, morphological operations on binary image masks, and some heuristics at the stage of fish detection. Finally, a preliminary discrimination between fish and no-fish objects is performed. Developed algorithm has been implemented as a working model and tested on benchmark data taken in the real environment.

Keywords Visual surveillance · Background model · Object detection · Object tracking

P. Forczmański (✉) · A. Nowosielski · P. Marczeski
Faculty of Computer Science and Information Technology, West Pomeranian University of Technology in Szczecin,
Żołnierska Str. 52,71-210 Szczecin, Poland
e-mail: pforczmanski@wi.zut.edu.pl

A. Nowosielski
e-mail: anowosielski@wi.zut.edu.pl

P. Marczeski
e-mail: pmarczeski@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_14

1 Introduction

1.1 Motivation

Automatic fish recognition is one of the most crucial problems to be taken into consideration during the environmental studies. Two main applications are emerging today. The first one is connected with fish industry. The second with the protection of endangered species.

Fish recognition is an important task in polyculture of several fish species in freshwater culture systems. Polyculture of several fish species is one of the practices of cultivating fish in ponds. Its advantages and drawbacks have been described and discussed many times in the literature (e.g., [1]). Fish species in extensive (big water reservoirs) and semi-intensive (ponds of a few acres) freshwater culture systems feed on different natural resources [1]. Higher growth rates are reported when species are grown together but there is the necessity to sort harvested fish [1]. Fish need to be graded not only by species but also by length and mass/size. European Community regulations are strict in that matter. Before fish are sold for human consumption, they must meet stringent mass and length requirements. We direct the reader to [2] for exemplary solution of the automated sorting system.

At this point it should be emphasized that manual processing of fish sorting is economically expensive. It is also a stressful and harmful operation to the fish not meeting the size requirements. After they are harvested with fishing nets and placed on the sorting table, they are rejected (by paid workers) as not ready for marketing and thrown back into the pond. This is a labor-intensive management operation [2].

Human activity has contributed significantly to the extinction of many animal species. This also applies to aquatic environments and fish. Many rivers and their tributaries have been baffled with weirs or dams that impede or completely prevent the migration of bienvironmental fish. For this reason, in many cases, last autochthonous fish population perished forever. New ecological recovery programs for ecosystems restoration are emerging. Programs involving the restitution of sturgeon or salmon impose the need for unblocking rivers through the construction of fish ladders. The surveillance of the migration by analyzing the image obtained in fish ladder can contribute to a reliable data collection necessary in the process of restitution of extinct species.

In the chapter, we present a preliminary works on fish detection and classification by means of computer vision methods. We assume that fish swimming freely in a water tank is observed by a standard video camera. The video stream is analyzed and regions of moving fish are detected. Further, they are discriminated between fish and no-fish objects. Finally, they are labeled and passed to the recognition stage, which employs a silhouette classifier.

The paper is organized as follows: First, some previous works related to the fish detection, counting, and classification are presented, followed by a system outline (background model, object detector, tracker, and then object classifier). Finally, some experimental results and summary are given. The work ends with some conclusions and future works.

1.2 Previous Works

The problem of monitoring fish activities has been present in the scientific literature for several years. It can be divided into two areas. The first one includes fish detection and counting, while the second is associated with fish spices recognition.

The task of fish detection and recognition in unconstrained environment can be solved with the following automatic techniques:

- video processing (e.g., [3]),
- stereovision (e.g., [4]),
- infrared sensor systems (e.g., [5]),
- hydroacoustic devices like sonars (e.g., [6]).

The other methods employ traps, electrofishing, nets and rods, as well as redd counting (disturbances in gravel caused by mating activities of some fish).

In this work, we focus on methods that employ video images obtained from cameras, processed with computer vision techniques, that enable a wide range of applications in fish recognition problem. It is because such methods can be applied in different environments, not regarding fish spices and other limitations. One serious limitation here, however, is the image quality under water. Stereovision allows 3D image acquisition. It requires a proper calibration and the aquatic environment may contain suspended sediments which limit the depth of view. Infrared sensors are used mainly in the problem of counting fish. Their disadvantage is the need to use in tight spaces (i.e., directing fish through narrow passages where fishes break a light beam while traveling along). Sonars, in turn, are used to detect shoals.

An interesting method of fish recognition is presented in [2]. Three species were sorted in pond water containing algae and suspended sediments. The recognition process is performed in a narrow channel which is additionally illuminated. The contour of the fish is described in the polar coordinate system. The most salient pixel belonging to the fish is considered to be the tip of its head (the mouth). This point is used as the origin of a polar coordinate system. This way, the contour description is independent of the orientation. Features obtained from principal component analysis (PCA) and partial least squares (PLS) form the contour signatures are used for fish classification. Besides this authors also utilize geometrical features. Some characteristic landmarks are located along the contours and their relative lengths form a feature vector. Features were compared using a minimum Mahalanobis distance classifier. The best results were reported for the combination of contour signature and geometrical features. The combined two-stage model achieved 95 % accuracy. However, it must be noticed that fish were classified in a narrow channel which restricted swimming perpendicular to the camera's view direction.

A system for automated fish sorting and counting in fishways is presented in [5]. Proprietary infrared fish silhouette sensor is used. As the fish swims through the scanning units, it obstructs the infrared beams [5]. Fish contour is then generated. Description of fish silhouettes utilize moment invariants and Fourier boundary descriptor. Nine simple characteristic features (e.g., area, perimeter, length, height, and other) are also considered. Three types of classifiers are used in multiclassifier combination approach (the majority voting): Bayes maximum likelihood classifier, learning vector quantization (LVQ), and multilayer perceptron (MLP).

Another example of recent works devoted to these problems is a paper [3]. Presented method of detecting, counting, and classifying fishes found in underwater video images employs Viola and Jones Haar-like feature object detection. The system was implemented on a field programmable gate array (FPGA) to increase performance. First step is a Haar Training procedure which calculates Haar-like features based on positive images containing cropped fishes and negative images containing cropped backgrounds and fishes of other species. Then an AdaBoost algorithm is used to create a "rejection cascade" classification. The effectiveness of this method working on a PC for a 16-stage classifier is 92 % with 3 % of false positives. Due to FPGA framework limitations, authors were unable to specify hit/miss ratio of this algorithm running on a field programmable gate array but they showed that, depending on level of FPGA parallelism and image's resolution, the performance can be improved even by 85 %.

In the paper [7], the authors describe a novel method of classifying fish species based on their color and texture using a multiclass support vector machine (MSVM) for detecting fish diseases. This was done by extracting color features, statistical texture features, and wavelet-based texture features and testing them in a LIBSVM software. The results showed that Bior4.4 wavelet filter in HSV color space was the best model for fish classification. Based on the best color input two one-against-one MSVMs were tested: directed acyclic graph MSVM (DAGMSVM) and voting-based MSVM (VBMSVM) resulting in DAGMSVM being the fastest and most accurate.

2 Proposed System

2.1 Assumptions

We describe an optical surveillance system aimed at fish detection and classification. It fits the visual surveillance and intelligent sensors areas. Typical visual surveillance system is aimed at gathering information about certain phenomena in order to execute or suggest certain actions, especially in case of situations dangerous to human life, health, or property, as well as in case of environmental hazard. Visual surveillance is often realized using a closed-circuit television system (CCTV) and is applied to maintain close observation of people, animals, and other objects. It is very often, that a CCTV consists of static camera (or cameras) aimed at one fixed point in space.

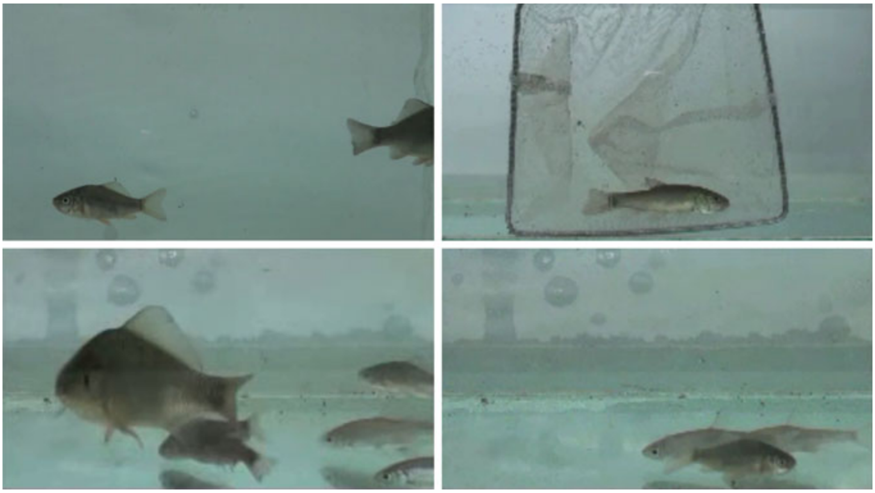


Fig. 1 Exemplary frames from the video streams used in experiments

The focal length of camera lens is constant as well. Such assumptions are true for an analyzed case of fish detection and classification. Exemplary frames from benchmark video taken by our simplified CCTV, showing the water environment and moving fish are presented in Fig. 1. Since most of scenes observed by the CCTV cameras are not static, the process of background separation has to take into consideration many different environmental conditions, such as variable lighting [8], atmospheric phenomena, and changes caused by different actions. Hence, background modeling is a crucial task and its efficiency determines the capabilities of the whole system. Recently, many methods of background modeling have been proposed. They are based on different principles, but all of them can be divided into two main categories: pixel-based and block-based approaches. The former class of methods considers each individual pixel in the image, while the latter analyzes an image decomposed into segments (often overlapping). For each pixel or segment, certain features are calculated and later used at the classification stage (into pixels or segments belonging to background and foreground). In the typical approach, each detected object is also tracked. It is often assumed that the movement is constant and the direction does not change in a considerable way [9–11]; however, it is disputable in a case of tracking living creatures like fish. The last stage of processing involves object recognition or classification. The selection of a method depends mainly on the object type and its invariant features. In a system presented in [12], each detected object is described by the mean area it occupies, while the authors of [13] detect humans using average human body proportions and size. These approaches are simple to implement and their application to fish detection and recognition is quite straightforward.

2.2 Algorithm Overview

Presented algorithm analyzes video sequence in order to find areas potentially occupied by fish in water tank. There have been several such algorithms proposed over last years; however, the problem is still valid. The main issue is reliable object classification and accurate tracking. A system presented in this paper automatically detects foreground objects, verifies, and classifies them. We distinguish between two classes of objects that can be present at the scene: moving fish and other objects, e.g., nets, air bubbles, etc.

Let us assume that a video sequence entered to the system consists of a set of frames covering stable background and moving objects in front of it. These objects (e.g., fish) move in a stepwise manner, which is an effect of reduced frame rate (often equal or less than 25 frames per second). In cases when preliminary assumptions about static camera cannot be fulfilled, the sequence should be cut into segments and analyzed independently.

Proposed system consists of four main components, namely adaptive *Background Model*, *Noise Removal*, *Object Tracker* and *Classifier*. The main task of the background model is background removal. Foreground regions are segmented from the present frame in respect to the previously learned background model. Sequence of detected foreground blobs are processed using a set of noise removal procedures and then sent to the Tracker, which updates tracked objects and labels them. During tracking, every newly created object is classified by the Object Classifier and its label is stored in Object Tracker. A simplified scheme of the proposed system is shown in Fig. 2.

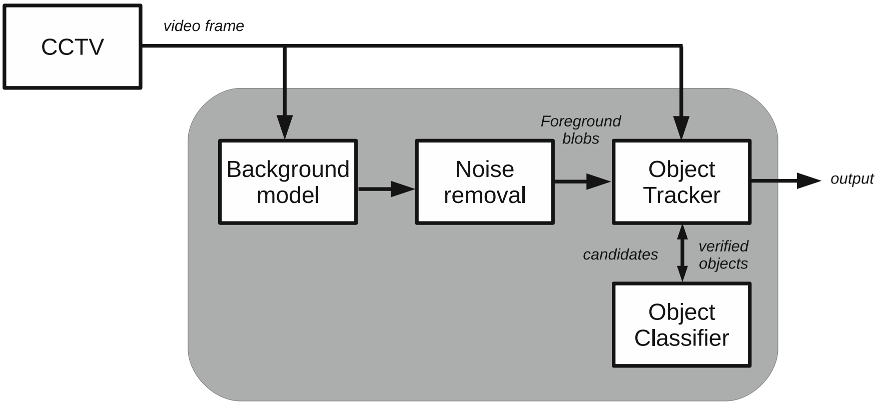


Fig. 2 Scheme of data processing in the proposed system

2.3 Background Model

The background model employs a pixel-based approach similar to the one proposed in [8]. In our case, every pixel is modeled by a set of five mixtures of Gaussians in R, G, and B channels. Such number of Gaussians distributions increases the robustness of the model in the comparison to the one presented in [14]. Similar approach, successfully employed for human motion tracking, was presented in [15].

The first 100 frames from video stream are used for learning the parameters of the background model. The further frames are processed in a stepwise manner, and the parameters of the model are updated.

During the processing loop, every pixel from the current frame is checked against the existing Gaussians in the corresponding position in the model. If there is no match, the least probable Gaussian is replaced by the new distribution using current pixel value as a mean. Then, the weights of all Gaussians are updated according to the following rule: weights of distributions that do not correspond with the new pixel value are decreased, while the weights of distributions that suite the new pixel value are increased. Parameters of unmatched distributions remain unaltered. The parameters of the distribution which matches the new observation are updated according to the following formulas:

$$\mu_t = (1-\rho)\mu_{t-1} + \rho X_t, \tag{1}$$

$$\sigma_t^2 = (1-\rho)\sigma_{t-1}^2 + \rho (X_t - \mu_t)^T (X_t - \mu_t), \tag{2}$$

$$\rho = \alpha \eta (X_t | \mu_k, \sigma_k), \tag{3}$$

where X_t is a new pixel value, η is a Gaussian probability density function, α is a learning rate, μ and σ are distribution parameters, and $\rho \in \langle 0, 1 \rangle$.

Afterward, each weight of each distribution is updated as follows:

$$\omega_t = \begin{cases} (1-\alpha)\omega_{t-1} + \alpha & \text{if a pixel fits the distribution} \\ (1-\alpha)\omega_{t-1} & \text{otherwise} \end{cases} \tag{4}$$

Background subtraction operation results in an image mask of possible foreground pixels which are grouped using connected components. Unfortunately, this approach does not suppress shadows and certain reflections from being considered as moving objects. Hence, it can cause certain serious problems, namely false detections of nonexistent objects. In the proposed system, we use a shadow detection and elimination method based on [16]. It employs an observation that casted shadow and darkens the point while chrominance of shaded and open regions does not vary much. In our approach, the algorithm detects shadows in Hue Saturation Value (HSV) color space which highly correlates with human perception of color. In order to detect shadow regions, for each pixel in the foreground regions we check the following conditions:

$$SP(x,y) = \begin{cases} 1 \ \text{if}\, \alpha \le \frac{I^V(x,y)}{B^V(x,y)} \le \beta \wedge (I^S(x,y) - B^S(x,y)) \le \tau_S \\ \quad \wedge |I^H(x,y) - B^H(x,y)| \le \tau_h \\ 0 \ \text{otherwise} \end{cases} \tag{5}$$

where SP is a new binary mask of a foreground blob, α, β, τ_h, and τ_s are empirically chosen parameters (according to heuristic approach described in [16]), I is a current frame, B is a background image and H, S, and V superscripts indicate HSV components, x and y are pixel coordinates in the image. Initial condition checks the first part of the assumption that shadow darkens the region, whereas next two conditions check if hue and saturation for shaded and open regions do not vary too much.

2.4 Noise Removal

Binary mask being a result of background modeling suffers from heavy noise. It comes from the fact, that water is an uneven environment and if often features changes of color, intensity, and clarity. Small particles moving in water also influence the background/foreground mask.

Hence, we perform a set of simple operations to remove isolated pixels or small groups of pixels that cannot be associated with a fish. The processing algorithm includes morphological closing and morphological opening, performed independently. The results are then aggregated using logical *and* operator. The result is a noise-free matrix. Exemplary images showing this procedure are provided below (see Fig. 3). After noise removal, we perform a heuristic-based operation that eliminates objects that are too small to be fish. We analyze the area of each independent object and if it is smaller than 1 % of whole image area (estimated as a multiplication of image's width and height), it is removed from further processing.

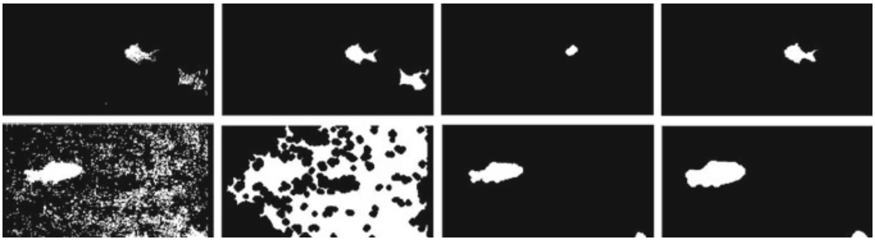


Fig. 3 Detected moving object before and after noise removal

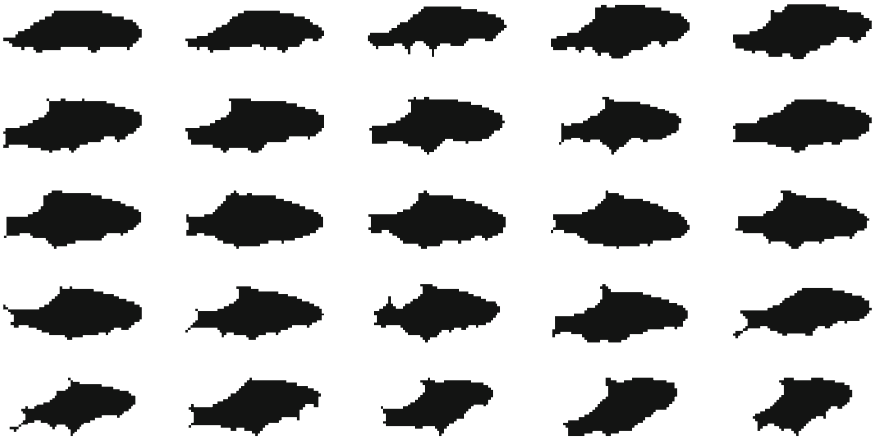


Fig. 4 Sample fish silhouettes extracted from video stream (from consecutive frames)

2.5 Object Tracker

In the next step, we track foreground objects using a simplified Object Tracker. Objects are tracked from frame to frame in a stepwise manner. For each tracked object (labeled using unique number), we store an information about its bounding box and its position in current frame. We uses historical data (previous frames) to decide about the object label. Besides such numerical data, the database contains, for each object, its binary mask (in each frame) and cropped video frame.

In order to match detected foreground blobs to tracked objects, we construct an association matrix similar to the one proposed in [17]. For all pairs of foreground blobs and tracked objects, we measure Euclidean distance from last recorded position of object to the center of the foreground blob. If foreground blob intersects with last remembered bounding box of the tracked object, we measure distance from the center of bounding box to the center of the blob. After distance calculation between all pairs blob-object, objects are updated using blobs which are closest to them. In case when a blob has no matched object, a new tracked object is created. On the other hand, when the object has not been associated to any foreground blob for several frames, it is removed. This case reflect situation when fish has left the scene. Exemplary results of fish tracking (fish silhouettes extracted from the foreground mask) are presented in Fig. 4.

2.6 Object Classifier

Most of the approaches aimed at silhouettes recognition presented in the literature use simple shape descriptors. An interesting comparative study was presented in [18].

Table 1 Accuracy (%) of different classifiers and features in terms of fish silhouettes recognition

	SSIG		FD	
Classifier	TP	FP	TP	FP
1-Nearest neighbor	57	31	65	22
Naive bayes	34	15	41	20
Simple logistic	57	43	62	36
Random tree	49	31	54	27
Random forest	63	30	72	23
Multi layer perceptron	56	29	61	25

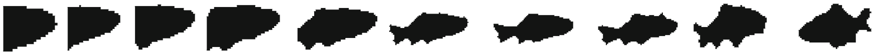


Fig. 5 Sample fish silhouettes extracted from consecutive frames that belong to the same class

We investigated two descriptors that are simple to calculate and proved to be effective in shape recognition tasks [19], namely *SSIG*—Shape Signature (based on calculating the distance from contour points to the centroid) and *FD*—Fourier Descriptor (calculated as a spectral transformation of contour points coordinates represented as complex numbers). The resultant vectors were normalized to a fixed number of points. In our experiments, the input data were fish silhouettes extracted from benchmark video (described below) in fully automatic manner. Sample silhouettes were shown in Fig. 4. It can be seen, that the silhouettes change over time, sometimes to a very high extent.

We preformed a set of investigations aimed at selecting the most suitable classifier. The comparison of the classifiers' performance are presented in Table 1. We focused on True Positive (TP) and True Negative (TN) rates as the most informative measures.

The main problem with recognizing (classifying) fish using their silhouettes is the nonrigid nature of their bodies (three dimensional and flexible characteristic). When we extract silhouettes from consecutive frames, the visual differences between neighboring frames is small enough, while it increases over time. The problem is depicted in the following figure (Fig. 5). That is the main reason of significantly low overall classification performance.

2.7 Experimental Results

Our experiments were performed on a set of five recordings taken in the large fish tank. Input video streams have been recorded in Full HD resolution (1920 × 1080) pixels and 30 fps (frames per second). In order to increase the performance of the software, we downscaled the frame size to 240 × 128 which occurred to be an acceptable resolution. The camera was operating in the full automatic mode (in terms of exposition and focus). The characteristics of the video clips are provided below:

- Clip #1: uniform background, fish swimming close to the bottom, visible reflections in the glass bottom, water movements, a net is sunken in the water;
- Clip #2: uniform background, fish swimming the whole tank, static water;
- Clip #3: uniform background, fish swimming outside a visible frame, water movements;
- Clip #4: uniform background, fish swimming close to the bottom, reflections in the bottom, static water;
- Clip #5: variable background, fish swimming close to the bottom, water movements.

The fish tank contains at least two fish spices of different sizes. Fish swim in any direction and can freely rotate around Z axis. It can be observed that fish often slow down or even stop for a while. Their speed of movement is variable and fish often occlude each other. Above observations make the detection and recognition quite a complex task.

The software implementing algorithms described above has been implemented in Matlab.

The performance of fish detection and extraction stage depends strongly on the content of the stream. In the simplest case, (Clip #2) the accuracy was equal to 75 %, while in more demanding cases, it falls down to about 50 % (Clip #1). During the tests we have observed the spurious detections in the glass wall of the tank, caused by moving nearby fish. Such misclassification problems were caused by small size and reduced contrast of those objects and can probably be solved with proper calibration of camera's exposure parameters. The performance of silhouette classification was presented above, in Sect. 2.6. The exemplary results of fish detection, segmentation, labeling, and tracking are presented in Fig. 6.

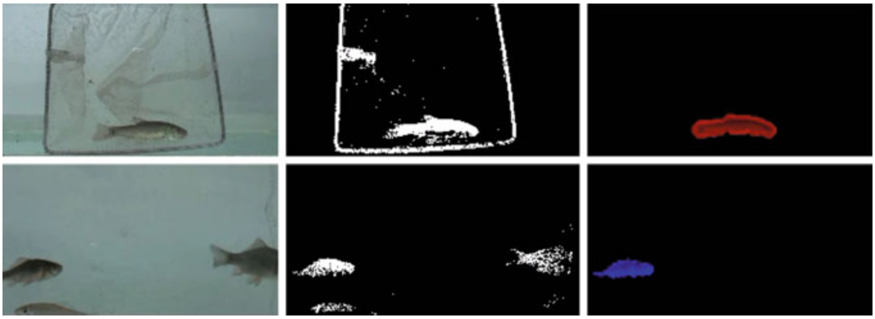


Fig. 6 Sample fish silhouettes extracted from video stream (from consecutive frames). In each row: original frame and foreground objects, binary fish mask, and fish marked in original frame

3 Summary

In the paper, we presented a model of a visual surveillance system aimed at fish detection and classification. Its field of application is associated with various environmental sciences. The presented algorithm incorporates adaptive background model, noise removal, object tracker, and a simple shape classifier. The developed method has been tested on video streams taken in laboratory environment and the chosen approach proved to be promising. The performance of adaptive background model is high; however, the accuracy of classification based on silhouettes is only acceptable. The accuracy of recognition may be improved by using other features like textures or movement characteristics. Hence this is the potential area of future works.

References

1. Zion, B., Shklyar, A., Karplus, I.: In-vivo fish sorting by computer vision. Aquac. Eng. **22**(3), 165–179 (2000)
2. Zion, B., Alchanatis, V., Ostrovsky, V., Barki, A., Karplus, I.: Real-time underwater sorting of edible fish species. Comput. Electr. Agric. **56**(2007), 34–45 (2007)
3. Benson, B., Cho, J., Goshorn, D., Kastner, R.: Field programmable gate array (FPGA) based fish detection using Haar classifiers. In: Pollock, N.W. (ed.) Diving for Science 2009. Proceedings of the American Academy of Underwater Sciences 28th Symposium. Dauphin Island, AL: AAUS (2009)
4. Ruff, B.P., Marchant, J.A., Frost, A.R.: Fish sizing and monitoring using a stereo image analysis system applied to fish farming. Aquac. Eng. **14**(1995), 155–173 (1995)
5. Cadieux, S., Lalonde, F., Michaud, F.: Intelligent system for automated fish sorting and counting. In: Proceeding of IEEE/RSJ, International Conference on Intelligent Robots and Systems (IROS), pp. 1279–1284 (2000)
6. Eatherley, D.M.R., Thorley, J.L., Stephen, A.B., Simpson, I., MacLean, J.C., Youngson, A.F.: Trends in Atlantic Salmon: the role of automatic fish counter data in their recording. Scottish Natural Heritage Commissioned Report No. 100 (ROAME No. F01NB02), Edinburgh, (2005)
7. Hu, J., Li, D., Duan, Q., Han, Y., Chen, G., Si, X.: Fish species classification by color, texture and multi-class support vector machine using computer vision. Comput. Electr. Agric. **88**, 133–140 (2012)
8. Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), pp. 246–252 (1999)
9. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. IEEE Trans. Pattern Anal. Mach. Intell. **25**(5), 564–577 (2003)
10. Welch, G., Bishop, G.: An introduction to the Kalman filter, Course 8, SIGGRAPH (2001)
11. Okarma, K., Mazurek, P.: Application of shape analysis techniques for the classification of vehicles. In: Mikulski, J. (ed.) Transport Systems Telematics: Communications in Computer and Information Science, vol. 104, pp. 218–225 (2011)
12. Li, L., Ma, R., Huang, W., Leman, K.: Evaluation of an IVS system for abandoned object detection on PETS 2006 datasets. In: Ninth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS), pp. 91–98. New York (2006)
13. Martinez del Rincon, J., Elias Herrero-Jaraba, J., Gomez, J.R., Orrite-Urunuela, C.: Automatic left luggage detection and tracking using multi camera UKF. In: 9th IEEE International Work-

shop on Performance Evaluation of Tracking and Surveillance (PETS), pp. 59–66. New York (2006)

14. Tian, Y., Feris, R.S., Hampapur, A.: Real-time detection of abandoned and removed objects in complex environments. In: IEEE International Workshop on Visual Surveillance (in conjunction with ECCV'08), Marseille, France (2008)
15. Forczmański, P., Seweryn, M.: Surveillance video stream analysis using adaptive background model and object recognition. Computer Vision and Graphics. LNCS, vol. 6374, pp. 114–121. Springer, Heidelberg (2010)
16. Cucchiara, R., Grana, C., Piccardi, M., Prati, A., Sirotti, S.: Improving shadow suppression in moving object detection with HSV color information. In: IEEE Intelligent Transportation Systems, pp. 334–339 (2001)
17. Lv, F., Song, X., Wu, B., Singh, V.K., Necatia, R.: Left-luggage detection using Bayesian inference. In: 9th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS 2006), pp. 83–90 (2006)
18. Zhang, D., Lu, G.: Review of shape representation and description techniques. Pattern Recognit. **37**(1), 1–19 (2004)
19. Frejlichowski, D., Forczmański, P.: General shape analysis applied to stamps retrieval from scanned documents. Artificial Intelligence: Methodology, Systems and Applications (AIMSA). LNAI, vol. 6304, pp. 251–260, Springer, Heidelberg (2010)

The Jitter Use to Reduce EMI on the Power Lines for Multi-core Processors

Tomasz Król, Leonard Rozenberg and Michał Twardochleb

Abstract In this paper, possibilities to reduce electromagnetic interference (EMI) in Globally Asynchronous Locally Synchronous (GALS) digital systems by applying jitter modeled by Monte Carlo simulation have been examined and presented. The aim is to protect digital systems which are prone to errors due to undesirable noises at current processors' miniaturization. By using a special software to analyze EMI, several different abstract models of GALS circuits have been designed in order to extract realistic data regarding their EMI characteristics. Based on the derived clock characteristics and our tool, we were able to analyze current profiles of each of the modeled systems, both in frequency and in time domain. The results were compared with their synchronous counterparts. Further investigations have shown that EMI reduction can be achieved by using low-noise GALS systems, in comparison with their synchronous equivalents. In addition, we have analyzed EMI reduction regarding granulation of the GALS system with different jitter settings. The results were extrapolated to asynchronous systems.

Keywords Monte Carlo · Jitter · GALS · EMI · Reduction

1 Introduction

Rapid and continuous development of digital circuits' production technology and its constant miniaturization expose their designer to new challenges. Traditional synchronous approach is now becoming very inflexible due to problems associated with

T. Król (✉), L. Rozenberg and M. Twardochleb
Department of Computer Science,West Pomeranian University of Technology,
Żołnierska 49, 71-210 Szczecin, Poland
e-mail: tomaszkrol85@gmail.com

L. Rozenberg
e-mail: lrozenberg@zut.edu.pl

M. Twardochleb
e-mail: mtwardochleb@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_15

time constraints and timing of the entire system. Even more demanding, regarding these aspects, are mixed digital–analog systems. Analog systems are very sensitive to noises generated by digital components. It can lead to data transmission errors, and in extreme cases even damage the tracks. Therefore, additional mechanisms must be used to reduce EMI (Electromagnetic Interference) at the maximum processor load formed during the simultaneous trigger of the whole clock domain. GALS systems (Globally Asynchronous Locally Synchronous) have been proposed as a new approach to integration of digital systems many years ago and now ready-made solutions are available [1, 13]. Previously, there were also few proposals of applying GALS methodology to reduce EMI [2]. Based on a few examples, it was shown that asynchronous systems can significantly reduce electromagnetic noise in comparison with their synchronous equivalents. Several solutions showing noise reduction using asynchronous methods have been described, including ARM processor design [3, 4]. There are also preliminary researches regarding the GALS systems application to reduce EMI. [2] It was shown that with GALS systems, EMI reduction can be achieved up to 20 dB, in comparison with their synchronous counterparts. In time domain, noise can be reduced up to 40 %. However, measurements on a real GALS system showed a smaller reduction of EMI. In addition, these activities were not systematic and focused only on specific cases, without taking into account GALS systems as a general methodology for system integration. Current technological advancement and further miniaturization of digital devices increase the demand for deeper research in this direction due to the detrimental effect of EMI on the entire system performance.

Due to an inability to identify behavioral determinants of the studied systems, analytical approach is unfeasible. This situation leads to selection of statistical modeling method, known as the Monte Carlo method, to investigate the issue of optimal selection of system parameters in order to reduce EMI.

2 Monte Carlo Method

According to the definition [5], the method of statistical tests (otherwise known as the Monte Carlo method) consists of seeking solutions to various problems by structuring numeric mathematical for each task stochastic process with parameters equal to the sizes of the searched task. Approximate values of these quantities are obtained by observations of stochastic process and calculation of statistical characteristics approximately equal to the searched parameters.

However, the idea of Monte Carlo method has been given a few decades ago, its practical use has become possible relatively recently, with development of computer technology. Only digital machines provide a sufficiently large computational power to perform an accurate statistical modeling of complex processes, which involves multiple recourses to series of mathematical computations. Moreover, use of pseudorandom number generators with a possibility to "take-snapshot" allows repeating experiments under the same conditions, which guarantee receiving identical results and overall comparability of experiments (in case of purely random generators, for example based on observation of a decay of atomic nuclei, the results obtained in following experiments, for obvious reasons, could not be the same. Additional difficulty is a slow process of modeling using these generators).

It should also be noticed that the most optimization problems and tasks are now being solved by computers. When solving a task of high computational complexity, the possibility of random errors occurrence (that may be caused by use of floating point notation, which is characterized by limited precision) must be expected. Therefore, the obtained solution is always considered as correct with a certain probability, of course, close to one [6]. Consideration of floating point notation, in a particular case may result in minimizing the differences between the degree of accuracy of the results obtained by methods of random sampling and classical methods accounting, solved using computers.

Thus, the Monte Carlo method allows finding solutions of complex problems through repeated testing of samples. Instead of looking for the results with an analytical method a random process can be designed and constructed. Observations of the results obtained with correspondingly large number of samples would effect with results close to selected.

Error ε of Monte Carlo method can be expressed as the difference between the actual observed value and the value received during the simulation. This error is inversely proportional to the square root of number of samples:

$$\varepsilon \sim \frac{1}{\sqrt{n}} \tag{1}$$

The scale of advantages resulting from radical simplification of calculation procedure allows acceptance of the results with minor mistake.

Particular areas in which statistical modeling methods can be used are topics which do not require the high precision of the result obtained, which is due to the structure of the problem consists in estimating the effects of possible solutions. Practical applications of this approach are described for example in [7].

3 Pseudorandom Jitter Generator

In previous papers [8, 9] it was shown how to reduce EMI in GALS systems. The results of experiments indicated that particular advantageous can be obtained by using jitter applied to clock source. Jitter introduces a phase modulation (sudden fluctuations of phase) to a clock wave at each clock cycle affecting the EMR (Electromagnetic Radiation—electromagnetic radiation) [10]. This modifies slightly the moment when the rising edge of the clock occurs. However, the average cycle and clock period remains unchanged. Thus, jitter may slightly increase or decrease the clock period of the cycle, but in general, the average base rate remains the same.

This mechanism was modeled and presented in Fig. 1. There are two major steps in generation of the jitter module. First, in order to avoid undefined states (glitches), and

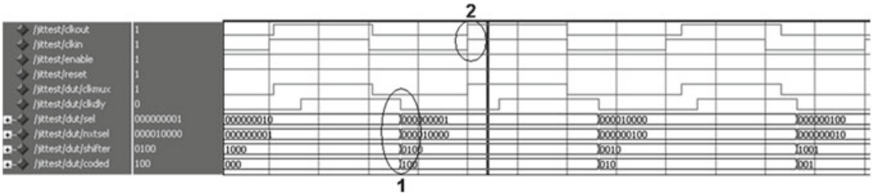


Fig. 1 Jitter simulation results

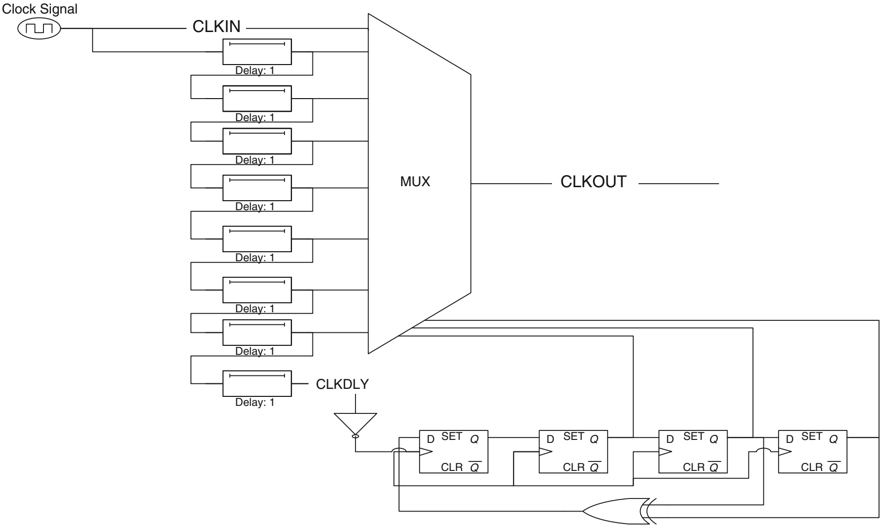


Fig. 2 Sample structure of jitter generator with LFRS 15 (*2^n 1 with n flip-flops of 4*)

to keep the length of the output signal, and additional signal has been introduced—CLKDLY. It is slightly more delayed than the last signal from the delay line (Fig. 2) and has a constant value in each clock cycle.

Therefore, when both signals—the input one (CLKIN) and the output one (CLKOUT) are in a low-state, there is a time slot for LFSR change (linear feedback shift register) and selection of a new delay line from multiplexer (step 1).

When CLKIN signal goes to a high-state (step 2), the multiplexer is already set to an appropriate delay line. It is also controlled by signals that come from the LFSR. In the example (Fig. 2) there is no need to use all four signals from LFSR to control multiplexer containing only eight delay lines. For this reason, only three signals are used to propagate the appropriate value. However, the 4-bit LFSR increases the randomness of generated delays, because as it is known, the longer linear feedback shift register the period of the generated sequence is longer and thus better imitates the real noise [11].

Table 1 presents the results of measurements of pseudorandom jitter generator at RTL level (register transfer level). They were determined by simulation, in order to

Table 1 Jitter generator—output signal delay

	Delay line	Delay (ps)
MIN	00000001	207
Delay	00000010	357
	00000100	479
	00001000	635
	00010000	751
	00100000	906
	01000000	1019
MAX	10000000	1175

obtain an accurate data of jitter to investigate its impact on the performance of the entire digital system. The input clock was set at 100 MHz (10 ns cycle time). "Delay Line" means a line that has been selected in current cycle. In addition, individual parameters were determined for each of the lines.

The second column presents the delay between the transition of the input clock and the output from low to high state. Despite the fact that the delay is increased with the increase of the delay line, it is not linear. The difference between the two values is not constant and it is caused by various path lengths that go to multiplexer. However, we can assume for further research, the average rate is 130 ps and is equal to the propagation time of two connected inverters.

It can be observed (Table 1) that using 100 MHz clock and 8-elements delay line, the maximum clock cycle reduction is 10 %. This is more than sufficient value for our study, because in practice and in the literature there are no examples of a successful implementation of a synchronous system with higher jitter parameters. On the other hand, if we would use 200 MHz clock source and applied four delay elements we would also get a 10 % jitter. The results would be similar with a slower clock source. Using a 50 MHz clock could increase the resolution of the generated fluctuations with the same delay elements (inverters) precision. It is also possible to replace the delay elements implemented as two connected inverters by using dedicated components (e.g., specially elongated tracks). It is assumed that the delay generated by such a construction would have shorter propagation time than two inverters in that technology. This would increase the resolution of jitter and its further ability to generate appropriate fluctuations at higher clock speeds.

During the simulations with modeled GALS systems it was noted that a change in jitter parameters significantly affects the level of EMI reduction; however, the change was not linear or uniform and depended on the characteristics of the GALS systems.

Due to the lack of observable determinants, Monte Carlo simulations were run to determine the optimal jitter parameters for different variants of GALS systems.

A set of three sample topology has been created for investigations (with different granularity) with configurable parameters (frequency of individual blocks, jitter parameters for each module, power consumption for the various components of the system). These topologies are shown in Fig. 3.

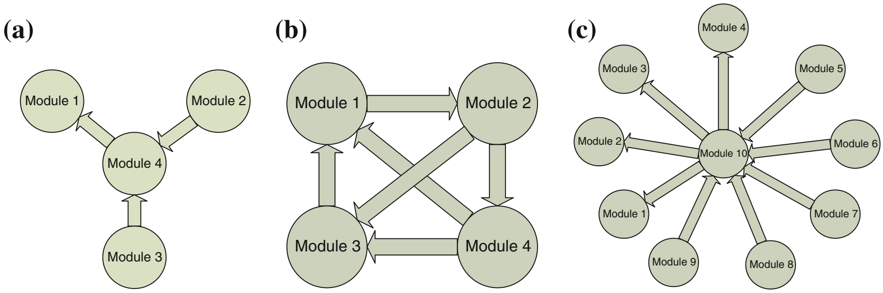


Fig. 3 GALS topologies: **a** star, **b** grid, **c** big star

4 Investigation Results

The simulations were conducted in special software designed merely to investigate electromagnetic interference in digital circuits. It is innovative software [12], whose efficiency and reliability of obtained results have been confirmed on actual digital circuits produced in laboratories of scientific research institute IHP in Frankfurt an der Oder (Germany). Additionally, all studies were performed with the same parameters of Monte Carlo method. Each simulation contained a population of 200 runs consisting of a series of randomly applied current waveform shifted in time by a random jitter value in the predetermined range. The probability of selecting one out of five current profiles was 20 %. Similarly, the probability of selecting a given value of jitter was uniformly distributed. Subsequently, all the waveforms were subjected to averaging in order to obtain consistent results. Each of the tests were conducted for 11 frequencies' ranges, in order to verify noise in wide spectrum and against different harmonics of the signal. To generate random sample an algorithm of "Mersenne twister" was used with a period of $2^{19937} - 1$.

4.1 10 Modules Synchronous System

First of the examined systems had a star topology, and consisted of 10 modules clocked at 50 MHz. A simulation was performed for the selected 11 frequency ranges with an assumption that jitter will modify the base frequency in the range from 0 to 15 % (Fig. 4).

It can be seen (Fig. 4.), that a significant reduction of EMI is visible in range of 1–10% jitter modification. Further there is observed no or very slight decrease of EMI reduction in all frequency bands. Additionally, it is evident that the lower frequencies are significantly less extinguished than the high ones. This is related to the fact that jitter much more effects the higher frequencies by implementing minor phase fluctuations.

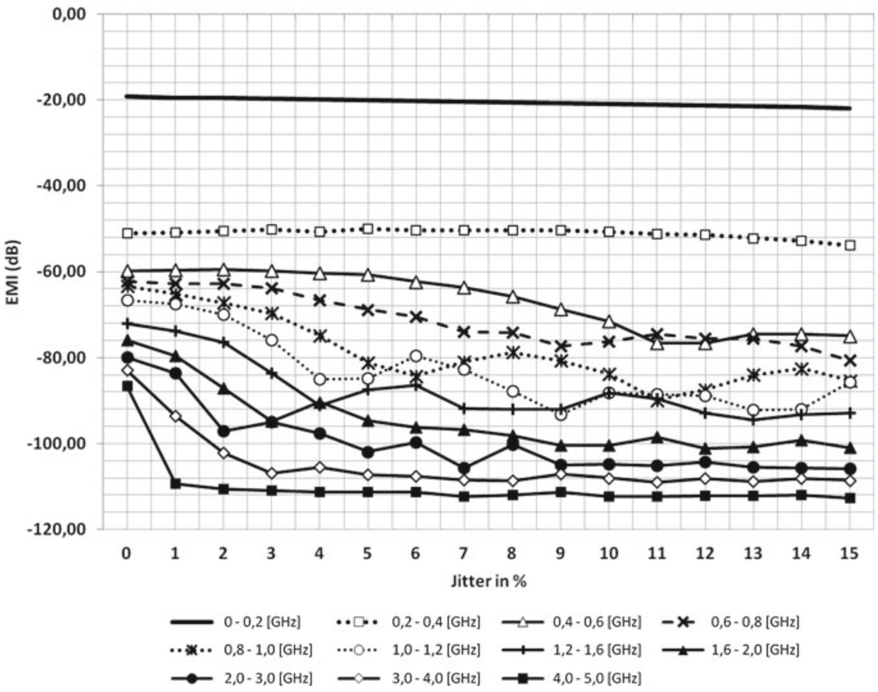


Fig. 4 Simulation results of 10 modules star topology synchronous system

4.2 Four Modules Synchronous System

Next system that was analyzed had the star topology, and consisted of four modules clocked at 50 MHz. As in the previous scenario, a simulation was performed for 16 settings of jitter from 0 to 15 %. Likewise the previous simulation, we can see (Fig. 5) that the suppress rate of a particular frequency depends on its range. Thus in the first place the higher ranges are muted. However, the lower ones have quite linear characteristic. This results directly from the fact that the lower frequencies would be much faster suppressed by for example 50 % jitter. However, it can be easily achieved by a phase shift of each block, which is not a topic of this experiment.

4.3 Four Modules Asynchronous System

In third experiment, a GALS system with four modules was analyzed. Each of modules had a different clock generator (46, 48, 50, and 52 MHz). We can notice (Fig. 6) that in the asynchronous GALS system, the lower frequencies do not undergo practically to any suppressing. This is due to the properties of such systems. These sys-

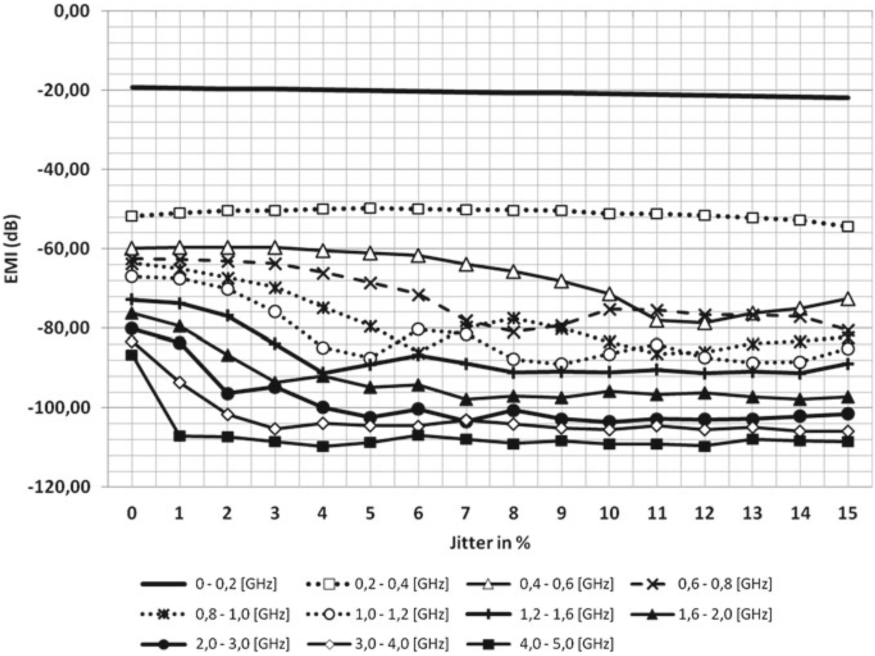


Fig. 5 Simulation results of 4 modules star topology synchronous system

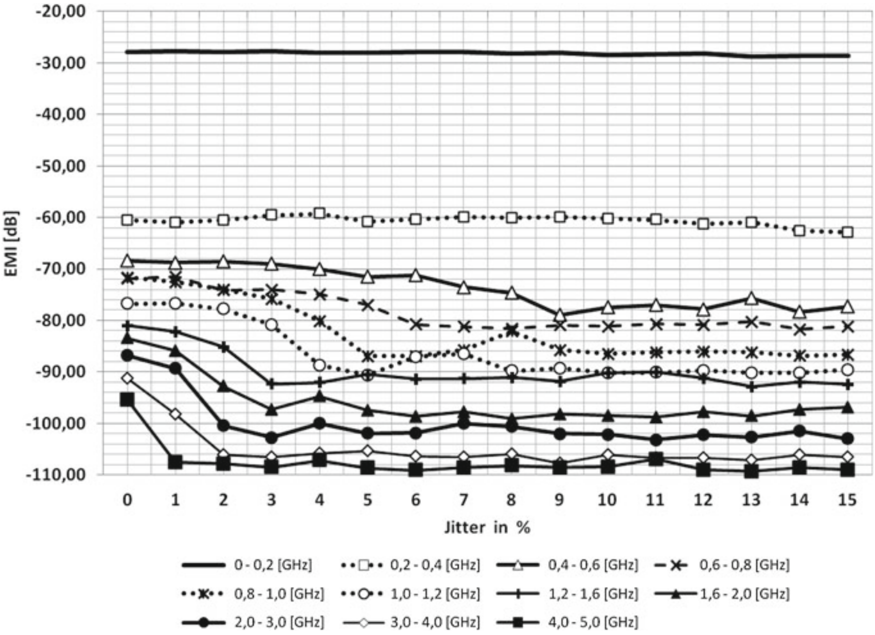


Fig. 6 Simulation results of 4 modules star topology asynchronous system

tems have stoppable clocks that generate by its nature phase shifts. This affects the dispersion of current peaks in time, thereby reducing noise in the lower frequencies. However, a higher frequency of about 500 MHz, experience similar reductions, as it was in case of synchronous systems. The higher the frequency, the harder and faster they are suppressed according to the jitter level.

5 Conclusion and Remarks

In this paper investigation results of EMI reduction in digital systems by jitter generated in modeled GALS systems are presented. The simulations were run according to Monte Carlo method. The borders of EMI reduction have been investigated in such systems. As a result, it has been shown that the optimum range of jitter parameters for different systems and different topologies are close to 10 %. Increasing this parameter as evidenced does not affect further significant reduction in EMI. Additionally, increasing of this parameter might have detrimental impact on a whole system by causing a number of errors during its operation. Furthermore, it was presented that the lower frequencies are significantly less, but more linearly reduced in terms of electromagnetic interference in synchronous systems. However, in the asynchronous GALS systems because of their characteristics, which are different blocks' frequencies, jitter impacts only higher frequencies. In further investigations, authors will try to use these results to investigate possibility of reducing electromagnetic interference in asynchronous systems using phase shift, which is by nature in GALS systems. The model will be extended for a further characteristic of electronic circuit structures. Hybrid optimization will be used to determine relevant parameters for digital system. In addition, one objective parameter based on the presented 11 frequency ranges, which the electromagnetic interference has been yet investigated in, will be defined.

References

1. Muttersbach, J., Villiger, T., Fichtner, W.: Practical design of globally-asynchronous locally-synchronous systems. In: Proceedings of the IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC), pp. 52–59. Eilat, Israel (2000)
2. Grass, E., et al.: Enhanced GALS techniques for datapath applications. In: Proceedings of PATMOS Workshop, LCNS 3728, pp. 581-590. Springer, Leuven (2005)
3. Furber, S.B., et al.: AMULET2e: an asynchronous embedded controller. Proc. IEEE **87**(2), 243–256 (1999)
4. Bink, A., York, R.: ARM996HS: The first licensable, clockless 32-bit processor core. IEEE Micro **27**(2), 58–68 (2007)
5. Buslenko, N.P., Metoda Monte Carlo, Buslenko, N.P., Golenko, D.I., PWN, Warszawa (1967)
6. Białynicki-Birula, I., Modelowanie rzeczywistości//Prószyński i S-ka, Warszawa (2002)
7. Twardochleb, M., Włoch, P.: Wybrane problemy i zastosowania. Wspomaganie procesu podejmowania decyzji dla modelu zagadnienia inwestycyjnego z wykorzystaniem symulacji Monte Carlo w: Technologia informacyjna. Uniwersytet Szczeciński, Szczecin (2010)

8. Blunno, I., Passerone, C., Narboni, G.A.: An automated methodology for low electromagnetic emissions digital circuits design. In: Proceedings of Euromicro Conference on Digital System Design (DSD), pp. 540–547. IEEE (2004)
9. Badaroglu, M., et al.: Clock-skew-optimization methodology for substrate-noise reduction with supply-current folding. IEEE Trans. CAD/ICAS **25**(6), 1146–1154 (2006)
10. Badaroglu, M., et al.: Digital ground bounce reduction by phase modulation of the clock. In: Proceedings Design, Automation, and Test in Europe (DATE) Conference, vol. 1, pp. 10088 (2004)
11. Balph, T.: Motorola Semiconductor. LFSR counters implement binary polynomial generators, EDN, Design feature
12. Krol, T., Krstic, M., Fan, X., Grass, E.: Modeling and reducing EMI in GALS and synchronous systems. Integrated Circuit and System Design, Power and Timing Modeling, Optimization and Simulation. Lecture Notes in Computer Science, vol. 5953, pp. 146–155. Springer, Berlin (2010)
13. Krstić, M., Grass, E., Stahl, C.: Request-driven GALS technique for wireless communication system. In: Proceedings of International Symposium on Asynchronous Circuits and Systems (ASYNC), pp. 76–85. IEEE, New York (2005)

A Study on Various Dimensionality Reduction Techniques Applied in the General Shape Analysis

Katarzyna Gościewska and Dariusz Frejlichowski

Abstract The paper presents selected numerical data dimensionality reduction techniques and their application to reduce the size of feature vectors used in an exemplary General Shape Analysis (GSA) task. The usability of applying reduced feature vectors was experimentally tested using three Fourier transform-based shape descriptors and three data reduction approaches. The aim of the experiments was to investigate which data reduction approach is the best, i.e., gives the highest percentage effectiveness value while maintaining a minimal size of the feature vector.

Keywords General shape analysis · Fourier descriptors · Feature dimensionality reduction

1 Introduction

The General Shape Analysis may be associated with traditional shape recognition or shape retrieval; however, shapes are recognized on a different detail level. Exact shape identification is not performed, only similarity between a test object and several general templates is established. In this approach, all shapes are represented using a particular shape descriptor and for each test object one or a few most similar general templates are indicated. The similarity or dissimilarity between shapes is estimated using various matching methods, e.g., the Euclidean distance or correlation coefficient. The GSA is a nontypical way of performing shape recognition. Unlike the traditional approaches, the processed shape does not have to belong to the base class. Such analysis may be useful when we are not interested in identifying a shape but finding a general class of it. Therefore, the GSA may be helpful in the case when the data are incomplete or there is no information about the type of the data.

K. Gościewska · D. Frejlichowski (✉)
Faculty of Computer Science and Information Technology,
West Pomeranian University of Technology, Żołnierska 52, 71-210 Szczecin, Poland
e-mail: dfrejlichowski@wi.zut.edu.pl

K. Gościewska
e-mail: kgosciewska@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_16

The GSA task has been already solved using various shape descriptors, such as the Point Distance Histogram [1] or Zernike moments [2], as well as relatively simple measures and shape ratios like the Feret diameters or Roundness [3]. One of the possible applications of the GSA approach is coarse classification, in which only those test objects which show greater similarity to one of the templates are further identified in more detail. The process of narrowing down of a group of objects can be performed iteratively and results in decreasing the computational time of algorithms used for recognition or identification of shapes. Another aspect is related to the reduction of feature vector size used for shape representation. This is particularly important in the case of reducing matching time and database size. There are several ways of decreasing the size of a shape descriptor. The first one is to use only a part of the original representation, e.g., a part of the coefficient matrix in the case of Fourier-based descriptors; however, this method does not ensure that the selected data contain all important information about a shape and are sufficient for shape discrimination. The second method for reducing data consists of the use of data dimensionality reduction techniques, such as the Principal Component Analysis or Discrete Cosine Transform. Both approaches enable the selection of a part of the original data that is characterized by a particular level of importance. In the case of PCA, eigenvectors representing the largest amount of information about data variability are obtained. In turn, DCT removes some of the higher frequencies, leaving lower ones that represent the most important and general shape features, which is simultaneously related to the main goal of the General Shape Analysis. A different approach to shape analysis can also be found in the literature. It was proposed by Rosin and concerns exploring shape characteristics using simple measurements designated to investigate, e.g., ellipticity, triangularity, or rectangularity of a shape [4–6].

The following sections contain a description of data reduction techniques and their usefulness in reducing feature vector size. The second section presents shape description algorithms which were used for solving the GSA task. These are algorithms based on the application of the Fourier transform, namely the Two-Dimensional Fourier Descriptor (2DFD), UNL-Fourier Descriptor (UNL-F), and Generic Fourier Descriptor (GFD). In the third section, PCA and DCT techniques are briefly explained. The fourth section contains a description of the experiments and experimental results for full and reduced shape representations. Results were considered correct if they coincided with the results provided by people in the inquiries. The last section summarizes and concludes the paper.

2 Fourier-Based Shape Descriptors

Shape description algorithms that use the Fourier transform for calculating the feature matrix or feature vector are relatively popular and commonly used due to the useful properties of the Fourier transform. As a result, transform-based shape representation algorithms are invariant to scale and translation within an image plane, and are robust to noise. The most basic approach consists of the use of the Two-Dimensional Fourier

Descriptor and a region shape—a representation has a form of a matrix with absolute spectrum values, which are obtained using the following formula [7]:

$$C(k,l) = \frac{1}{HW} | \sum_{h=1}^{H} \sum_{w=1}^{W} P(h,w) \cdot \exp^{(-i\frac{2\pi}{H}(k-1)(h-1))} \exp^{(-i\frac{2\pi}{W}(l-1)(w-1))} |, \quad (1)$$

where:

H, W—height and width of the image in pixels,
k—sampling rate in vertical direction ($k \geq 1$ and $k \leq H$),
l—sampling rate in horizontal direction ($l \geq 1$ and $l \ \leq \ W$),
$C(k,l)$—value of the coefficient of discrete Fourier transform in the coefficient matrix in k row and l column,
$P(h,w)$—value in the image plane with coordinates h, w.

The calculation and matching of Fourier-based descriptors are simple. This approach enables the generalization of information due to the fact that the most general shape features are stored in the left top corner of a coefficient matrix. This confirms the relevance of using only a part of the original full representation.

The UNL-Fourier Descriptor is based on the UNL (named after Universidade Nova de Lisboa) descriptor and Two-Dimensional Fourier Descriptor. The UNL uses a complex representation of Cartesian coordinates for points and parametric curves in a discrete manner [8]:

$$z(t) = (x_1 + t\,(x_2 - x_1)) + \mathrm{j}\,(y_1 + t\,(y_2 - y_1)), t \in (0,1), \quad (2)$$

where $z_1 = x_1 + \mathrm{j}y_1$ and $z_2 = x_2 + \mathrm{j}y_2$ are complex numbers. The shape centroid O is calculated using the following formula [8]:

$$O = (O_x, O_y) = \left(\frac{1}{n} \sum_{i=1}^{n} x_i, \frac{1}{n} \sum_{i=1}^{n} y_i \right), \quad (3)$$

where n is a number of points in a contour and x_i, y_i are Cartesian coordinates of the ith point. Next, the maximal Euclidean distance between contour points and centroid is found [8]:

$$M = \max_i \{ \|z_i(t) - O\| \} \quad \forall i = 1, \ldots, n \quad t \in (0,1). \quad (4)$$

Ultimately, a discrete version of the new coordinates is derived as follows [8]:

$$U(z(t)) = \frac{\left\| \left(x_1 + t(x_2 + x_1) - O_x \right) + \mathrm{j}(y_1 + t(y_2 + y_1) - O_y \right) \|}{M} + \mathrm{j} \times \arctan \left(\frac{y_1 + t(y_2 - y_1) - O_y}{x_1 + t(x_2 - x_1) - O_x} \right). \quad (5)$$

The parameter t is discretized in the interval [0, 1] and new coordinate values are put into a matrix, in which rows represent distances from the centroid, and columns represent the corresponding angles. As a result, a Cartesian image containing an unfolded shape contour, as can be seen in polar coordinates, is obtained. A two-dimensional UNL representation enables the calculation of the Two-Dimensional Fourier transform.

The last shape description algorithm is the Generic Fourier Descriptor. It is similar to the UNL-Fourier Descriptor, with the only difference that it uses the region shape. In the first step, all pixel coordinates of an image are transformed into polar coordinates, and new values are put into a rectangular Cartesian image, in which the row elements correspond to distances from centroid, and the columns represent 360 angles in degrees [9]. The second and last step is the calculation of Fourier coefficient matrix. All the mentioned shape descriptors are calculated for two-dimensional images, therefore the number of Fourier coefficients is equalled to the number of pixels in an input image. The obtained representations are relatively large, containing 40,000 elements for 2DFD, 36,000 elements for GFD, and 10,000 elements for UNL-F. The necessity for reduction is therefore obvious. It may be achieved by selecting a part of the original spectrum or using data reduction techniques that are presented in the following section.

3 Selected Data Dimensionality Reduction Techniques

Undoubtedly, minimizing the time and computational complexity of algorithms is an important issue. It is particularly significant when the data collected are processed and compared several times, as well as in the case of object recognition or searching large multimedia databases. As a result, it is, on the one hand desirable to reduce the size of shape representations and, on the other, decreases the number of objects subject to processing. In machine learning, dimensionality reduction consists of reducing the number of analyzed variables—a low-dimensional representation of multidimensional data is obtained. Two groups of methods can be distinguished in this field—supervised and unsupervised learning-based methods, and methods for feature selection and transformation. Furthermore, transformation methods can be divided into feature extraction and feature generation techniques. In analyzing the GSA task, which is similar to pattern recognition, it is necessary to use feature extraction methods, such as the Discrete Cosine Transform, Principal Component Analysis, or Linear Discriminant Analysis (LDA) [10]. Due to the fact that the GSA approach is not aimed at classification, but only assigning several templates to each test object, LDA cannot be applied because it is a supervised learning-based technique. Attention will be focused only on PCA and DCT.

The Principal Component Analysis is an unsupervised, linear dimensionality reduction technique. It enables the construction of a low-dimensional data representation describing the greatest variability of the original data. To reduce data dimension using PCA, we have to find linear combinations of the original variables, which will be uncorrelated and are characterized by the highest variance—these combinations are principal components. As a result, only a small number of first components containing the most varied data are left and the rest is removed, still the information loss is small. The following description is based on [11, 12].

In the discussed approach, a matrix of feature vectors is used as input for PCA. Each row corresponds to one feature vector that is a Fourier descriptor matrix transformed into a vector. As a result, the number of rows is kept, and the number of columns is reduced to the number of expected principal components. In the next step, for every row i, an average $u[m]$ is calculated and subtracted from every value in the corresponding row giving a deviation matrix M. The following formulas are used:

$$u[m] = \frac{1}{N}\sum_{n=1}^{N}[m, n], \tag{6}$$

$$M[i, j] = X[i, j] - u[i], \tag{7}$$

where N is the number of elements in matrix. Subsequently, a covariance matrix C is derived as follows:

$$C[p, q] = \frac{1}{N}\sum M[i, j] \cdot M'[i, j]. \tag{8}$$

Next, based on the covariance matrix, eigenvectors and eigenvalues are obtained:

$$V^{-1}CV = D, \tag{9}$$

where V is a matrix with eigenvectors and D is a diagonal matrix containing eigenvalues. Pairs of corresponding eigenvectors and eigenvalues are sorted according to the decreasing eigenvalues. Eigenvectors with the highest eigenvalues are left and the remaining ones are removed giving the data reduction. The remaining eigenvectors are combined in a feature vector, which actually is a matrix with eigenvectors stored in columns. In the last step, a new set of data are obtained as a result of the product of input data and feature vector. The resulting matrix contains a number of columns which are equal to the number of selected eigenvectors, but the number of rows remains the same.

The use of DCT in pattern recognition results in the removal of high frequencies corresponding to details in images [13]. Similarly to the Fourier transform, the cosine transform also performs conversion to the frequency domain, however it uses only real numbers. DCT is a linear, reversible function and its most popular variant is DCT-II. It converts a finite number of data X_n into a sum of cosine functions oscillating at various frequencies. The C_k coefficients are derived using the following formula [14]:

$$C_k = \sum_{n=0}^{N-1} X_n \cos\left[\frac{\pi}{2}\left(n + \frac{1}{2}\right)k\right], \tag{10}$$

where $k = 0, 1, \ldots, N-1$. The presented variant of the transform is adequate for the GSA task, because shape descriptors will be processed as vectors. Most of the information about the signal is contained in a small number of low-frequency DCT coefficients. Dimensionality reduction is thus achieved by removing the highest frequencies, which contain a small amount of information [14].

4 Experimental Conditions and Results

In order to investigate the effectiveness of various forms of Fourier-based shape descriptors, several experiments concerning the GSA task with an exemplary shape database were performed. Four different representations that were obtained were subsequently compared using each shape description algorithm, i.e., full original representation, 2 × 2 subpart of the original representation, four PCA components, and four DCT components. In order to solve the GSA task in the first experiment, several steps were applied: all shapes from the database (see Fig. 1) were represented using the same shape description algorithm. Next, each test object representation was matched with all template representations to indicate three templates with the highest similarity. For matching purposes, the Euclidean distance was selected—the least distant objects are considered to be the most similar. In the next step, the coincidence between experimental results and results provided by people using inquiry forms was established. An experimentally indicated shape was considered proper if it coincided with one of three firstly selected shapes by people. In the second experiment, similar steps were applied but only a 2 × 2 subpart of the original representation was used. In the third and fourth experiment, an additional step of data reduction was added, which succeeded the calculation of shape descriptors and preceded the matching step.

The discussed experiments were repeated for each of the three shape descriptors. The results of the experiments in terms of percentage effectiveness are provided in

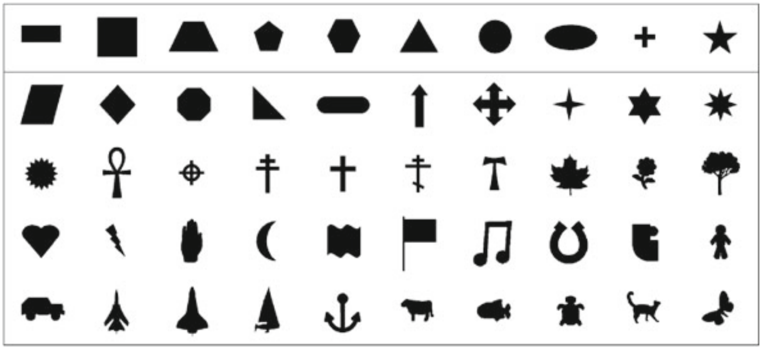


Fig. 1 Shapes used in the experiments (200 × 200 pixel size binary images)—10 templates (in the *first row*) and 40 test objects (*below*)

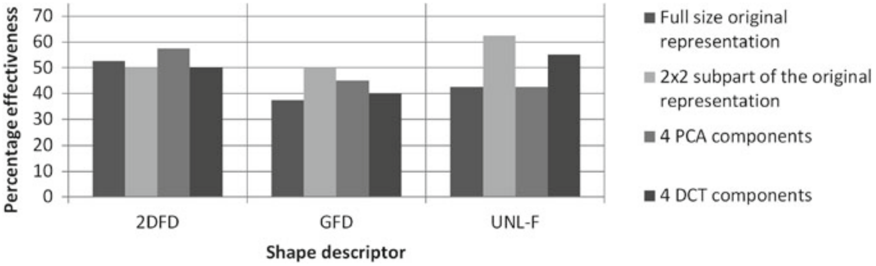


Fig. 2 Percentage effectiveness values obtained in the experiments using various shape descriptors

Fig. 2. The best result characterized by the highest effectiveness value and the smallest size of shape descriptor was obtained when a 2 × 2 subpart of the UNL-Fourier descriptor was used.

Generally, the results are not unequivocal, because it is not obvious which reduction variant could be considered as the best solution. However, there are some apparent dependencies. When comparing the original representation with the reduced representations of the Two-Dimensional Fourier Descriptor it can be seen that improvement is present only when the PCA is applied. In other cases, i.e., when a smaller subpart or DCT coefficients were used, the percentage effectiveness was lower than the full original representation. Despite small differences the result is very satisfactory, because large data reduction was achieved—from 40,000 elements to just four elements in the feature vector, while maintaining a similar level of effectiveness. In the experiments carried out with the use of reduced GFD and UNL-F representations, the percentage effectiveness values were equal or higher than those obtained using the original representation. Unfortunately, in the case of GFD and UNL-F, the dimensionality reduction techniques did not provide as good results as when the 2 × 2 subpart was used.

The above considerations may lead to the conclusion that some shape descriptors, such as GFD and UNL-F, which are invariant to affine shape transformations within an image plane, enable the effective reduction of the original representation by selecting only some of the first coefficients, hence the dimensionality reduction step using PCA or DCT is redundant. In turn, for the Two-Dimensional Fourier Descriptor, which is not rotation-invariant, the use of four principal components proves to be more beneficial.

Additionally, according to the tests presented in [15], all experiments were repeated with the use of the correlation coefficient instead of the Euclidean distance, however, all reduced representations gave worse effectiveness values comparing to the full size representations. Therefore, it can be concluded that in certain cases the use of specific dimensionality reduction approaches is justified, however both shape description algorithms and matching methods should be selected appropriately. Even if the increase in effectiveness is not too big, large data compression can be obtained.

5 Summary and Conclusions

The paper demonstrated the use of dimensionality reduction methods in the reduction of feature vectors calculated using various Fourier-based shape descriptors for the purpose of the General Shape Analysis task. The experimental results were compared with the results provided by people in order to evaluate the coincidence between the two, and obtain percentage effectiveness values. Three different shape description algorithms were used, namely the Two-Dimensional Fourier Descriptor, Generic Fourier Descriptor, and UNL-Fourier Descriptor, as well as four various forms of shape representation—full original representation, 2×2 subpart of it, four PCA components, and four DCT components.

The best solution for the General Shape Analysis task turned out to be the application of UNL-Fourier Descriptor and 2×2 subpart of the original representation. In this case, the use of PCA and DCT was redundant, however it did not worsen the effectiveness values compared to the original representation. In turn, the use the Two-Dimensional Fourier Descriptor benefited from the PCA-based reduction and the effectiveness value increased. Undoubtedly, there is some future work to be done, e.g., carrying out experiments using other shape description algorithms or different size of feature vectors in order to confirm the above-mentioned conclusions.

References

1. Frejlichowski, D.: Analiza Ogólnego Ksztatu Obiektów Wydobytych z Obrazów Cyfrowych Rozpoznawanych z Użyciem Deskryptora PDH. Metody Informatyki Stosowanej **1**, 5–13 (2009)
2. Frejlichowski, D.: The application of the Zernike moments to the problem of general shape analysis. Control Cybern. **40**(2), 515–526 (2011)
3. Frejlichowski, D.: An experimental comparison of seven shape descriptors in the general shape analysis problem. In: Campilho, A., Kamel, M. (eds.) ICIAR 2010, Part I. LNCS, vol. 6111, pp. 294–305. Springer, Heidelberg (2010)
4. Rosin, P.L.: Measuring rectangularity. Mach. Vis. Appl. **11**, 191–196 (1999)
5. Rosin, P.L.: Measuring shape: ellipticity, rectangularity, and triangularity. Mach. Vis. Appl. **14**, 172–184 (2003)
6. Rosin, P.L.: Computing global shape measures. In: Chen, C.H., Wang, P.S.P. (eds.) Handbook of Pattern Recognition and Computer Vision, 3rd edn., pp. 177–196. World Scientific Publishing Company Inc. (2005)
7. Kukharev, G.: Digital Image Processing and Analysis. SUT Press, Stettin (1998)
8. Rauber, T.W.: Two-dimensional shape description. Technical report: GR UNINOVA-RT-10-94, Universidade Nova de Lisboa, Lisoba, Portugal (1994)
9. Zhang, D., Lu, G.: Shape-based image retrieval using generic Fourier descriptor. Signal Process.-Image **17**(10), 825–848 (2002)
10. Cunningham, P.: Dimension Reduction. Technical Report UCD-CSI-2007-7 (2007)
11. Orfanidis, S.J.: SVD, PCA, KLT, CCA, and all that. http://eceweb1.rutgers.edu/~orfanidi/ece525/svd.pdf
12. Fodor, I.K.: A Survey of Dimension Reduction Techniques, http://computation.llnl.gov/casc/sapphire/pubs/148494.pdf

13. Sahoolizadeh, H., Heidari, Z., Dehghani, H.: A new face recognition method using PCA, LDA and neural network. In: Proceedings of World Academy of Science, Engineering and Technology (2008)
14. Denkowski, M., Mikołajczak, P.: Przetwarzanie obrazów cyfrowych—laboratorium. Instytut Informatyki UMCS, Lublin (2011)
15. Frejlichowski, D., Gościewska, K.: Application of 2D Fourier descriptors and similarity measures to the general shape analysis problem. In: Bolc, L., Tadeusiewicz, R., Chmielewski, L.J., Wojciechowski, K. (eds.) ICCVG 2012. LNCS, vol. 7594, pp. 371–378. Springer, Heidelberg (2012)

Knowledge-Based Approach to COTS Software Selection Processes

Agnieszka Konys

Abstract The process of COTS (Commercial Off-The-Shelf) software selection is difficult due to the large number of existing COTS components. A complexity of COTS selection processes has great impact on the COTS method development, (information tools and frameworks in consequence). The main problem is how to use it on a given decision problem, and which suits the best to COTS component selection processes. In this paper an analysis of selected COTS frameworks and information tools supporting COTS component selection processes is proposed. It is a basis for ontologies construction using OWL (Ontology Web Language) standard. The two separated COTS ontologies are presented in this paper: ontology for frameworks and ontology for information tools supporting COTS software selection processes. The general aim of this is to provide the knowledge-based approach to COTS software selection processes.

Keywords COTS · Ontology · COTS selection process · COTS frameworks and information tools

1 Introduction

COTS (Commercial-Off-The-Shelf) products play an important role on the market. Their practical application encompasses both engineering solutions (e.g., commercial tools for software development) and a wide spectrum of applied solutions. COTS are defined as ready to sell products, available in many copies with minimal changes [1]. COTS can be integrated with many different information systems. Additionally, they can be a part of bigger and more complex systems called COTS-Based System (CBS) [2]. The expected profits encompass time reduction of system construction and development and cost reduction sustaining a quality.

A. Konys (✉)
Faculty of Computer Science and Information Technology, West Pomeranian
University of Technology in Szczecin, Żołnierska 49, 71-210 Szczecin, Poland
e-mail: akonys@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_17

The COTS market offers a broad scope for COTS software products that support an enterprise in different domains. The popularity of COTS components and a huge number of them can cause data and information redundancy for a decision maker. The process of knowledge acquisition about COTS components is time-consuming, limited by restricted access to component information and documentation. A decision maker has to search and look through many sub-sites to find relevant information. Information provided by a vendor is subjective, oriented only on the strengths offered by a given solution. Traditional searching mechanisms (e.g., Google) do not include specified COTS functionalities, and they provide an incomplete set of results [3].

This is the main reason that large quantities of methods, frameworks, and information tools exist, and are still being developed and improved. The COTS methods provide only methodological (theoretical) aspects, while information tools and frameworks offer a wide spectrum of usage. Each of them has its own specification and narrow area of practical usage. For example, frameworks STACE and CRE are recommended to use for nonfunctional requirements evaluation. The application of framework CAP is based on using pre-existent experience. Furthermore, one of the considered information tools, MoReCOTS, is based on automatic components identification, and allows to identify components from Web resources. Other information tool like GOThIC provides a separated taxonomy [4–8].

An analysis of the literature does not provide any information about existing ontology for frameworks or information tools supporting COTS components selection processes [9–11]. It is postulated that ontology for frameworks and ontology for information tools supporting COTS component selection can reduce some of the identified research problems. The adaptation of knowledge-engineering mechanisms should improve the process of knowledge acquisition of COTS components and related selection methodologies.

The possible alternatives to knowledge acquisition of COTS (such as COTS component repositories, semantic techniques, independent reports or expert knowledge, etc.) are still in development phases. Some COTS repositories or semantic techniques are in prototype stages. It has been observed that COTS component repositories available on the market are excessively general solutions and it is difficult to use them for a specified domain. Other alternatives for knowledge acquisition of COTS are semantic search engines, but some of them do not contain specified information about the COTS domain (Table 1).

Table 1 Selected knowledge acquisition sources

Knowledge acquisition sources	Name
COTS software repositories (3)	CeBase COTS lessons-learned repository, CLARiFi, COTS components trading (COTSTrader)
Semantic techniques (10)	ONTOMANAGER, SymOntoX, Hierarchical Agglomerative Clustering (HAC), PLIB, INSEAS, RASCAL—Users web mining, Sema-SC (Semantic component selection), Semantic-based technique, NFR, MoreCOTS
Selected semantic search engines (4)	Swoogle, iGoogr, Yahoo SearchMonkey, WolframAlpha

In this paper a part of a complex knowledge-based approach to COTS software selection processes is proposed. The knowledge-based approach to COTS software selection is composed of four separated ontologies: ontology for methods and techniques [12], ontology for information tools, ontology for frameworks (provided in this paper), and ontology for COTS ERP components. The general aim is to provide a complex approach to support the COTS software selection processes. Three ontologies (for methods and techniques, information tools, and frameworks) provide methodological approaches to COTS software selection processes, whereas the ontology for COTS ERP components helps in component selection processes, and provides a specified knowledge of available COTS ERP components.

In this paper, two separate ontologies are presented: the ontology for frameworks and the ontology for information tools supporting COTS software selection and evaluation processes. A basis for ontology construction is an analysis of 15 frameworks (first ontology) and 24 information tools (second ontology). It is worth to notice that the knowledge of frameworks and information tools is scattered. The first ontology encompasses a set of 15 selected COTS frameworks, and the second ontology includes 24 selected COTS information tools. They are built using the Protégé application. Practical examples of the author's domain ontologies are provided. Moreover, consistency of the ontologies is checked.

2 Related Works

There are many approaches for COTS software selection that apply dissimilar methodologies in supporting the same software evaluation process. A large number of existing approaches emphasize the important role of a proper selection of software components. In the literature terms like methods and techniques, information tools, and frameworks exist as separate entities. Based on the literature, a framework is defined as a tool that supports testing, construction, and development of software application. A framework for COTS provides information about classification processes supporting COTS component selection, and presents the major issues about COTS component selection. Based on the literature, an information tool (supporting COTS component selection) is very often defined as a set of guidelines and applications for domain experts and end-users. Frameworks and information tools are developed to help in solving problems related to COTS software selection. Moreover, they ensure clearness and recurrence of software evaluation processes.

An analysis of the literature allows to identify 15 COTS frameworks (OTSO, STACE, PORE, CAP, CRE, CEP, FCS, CBCPS, ISO 9126, Morisio and Torchiano, Torchiano and Jaccheri, Delta Technology Framework, SSEF, Carney and Long, Carney and Wallnau [2, 5, 13–17]) and 24 information tools (ADIPS, Agora, Cognitive Task Analysis, CompoNex Browsing, EKD, GAM, GBRAM, GOThIC, GQM, HAC, INSEAS, IPSCom, MDA, OntoManager, PLIB, RASCAL, SCB, SDM, SYBIL, SemaSC, Semantic-Based Techniques [8, 18–28] (Tables 2 and 3).

Table 4 (Appendix 1) presents a synthetical comparative analysis of selected frameworks. The selected frameworks are based on technical and nontechnical

Table 2 The relations between selected COTS frameworks and the defined set of criteria

Name of criteria	Name of sub-criteria	CAP	CBCPS	CEP	CRE	Carney & Long	Carney & Wallnau	Delta Technology Framework	FCS	Framework ISO 9126	Morisio & Torchiano	OTSO	PORE	SSEF	STACE	Torchiano & Jaccheri
Evaluation	Evaluation of financial aspects	-	-	yes	-	-	-	-	-	-	-	yes	-	-	-	-
	Evaluation of quality aspects	-	-	-	-	-	-	-	-	yes	-	yes	-	-	-	yes
	Evaluation of attributes	-	-	-	-	-	-	-	-	-	yes	-	-	-	-	-
	Evaluation of product	yes	yes	yes	yes	yes	yes	-	yes	yes	-	yes	yes	yes	yes	-
	Evaluation of social-technical aspects	-	-	-	-	-	-	-	-	-	-	-	-	-	yes	-
	Evaluation of technology	-	-	-	-	-	-	yes	-	-	-	yes	-	yes	-	-
	Evaluation of non-funcional requirements	-	-	-	yes	-	-	-	-	-	-	-	-	-	yes	-
Evaluation process	Verification of software characteristics	-	-	-	-	yes	yes	-	-	-	-	-	-	-	-	-
	Supporting of attributes selection process	-	-	yes	-	-	-	-	-	yes	yes	-	-	-	-	yes
	Using of expert information	-	-	-	-	-	-	-	yes	-	-	-	-	-	-	-
	User attendance in a selection process	-	-	-	-	-	-	-	-	-	-	-	-	-	yes	-
	Using of requirements engineering techniques	-	-	-	-	-	-	-	-	-	-	-	yes	-	-	-
	Using of pre-existent experiences	yes	-	yes	-	-	-	-	yes	-	-	-	-	-	-	-
Requirements	Requirements analysis	yes	-	yes	-	-	yes	-	-	-	-	-	yes	-	yes	-
	Requirements-driven approach	-	-	-	yes	-	-	-	-	-	-	yes	-	-	-	-
	Supporting of requirements specification process	yes	-	-	-	-	-	-	-	-	-	-	yes	-	-	-
Usage	Additional supporting metrics	yes	yes	yes	-	yes	-	yes	yes	yes	yes	yes	yes	-	-	yes
	Attribute organizing process	-	-	-	-	-	-	-	-	yes	yes	-	-	yes	-	yes
	Systematization of software selection process	-	-	-	yes	-	-	yes	-	-	yes	-	-	-	-	-
	Using of additional applications	-	-	-	-	-	-	-	yes	-	-	-	-	yes	-	-
	Process reccurence	-	yes	-	yes	-	-	-	-	-	-	yes	yes	-	-	-
	Process systematization	-	yes	-	yes	-	-	-	-	-	-	yes	yes	-	-	-

approaches. Some of them (OTSO, STACE, PORE, CAP, CRE, CEP, FCS, CBCPS) exist in the literature as COTS methodologies.

The major part of analyzed COTS frameworks is based on software technology evaluation (Delta, SSEF, OTSO) [1, 13, 14]. Most of them require from a decision maker to predefine processes and criteria templates (CAP, PORE, CRE, CEP) [4, 6, 7, 29]. A part of the analyzed frameworks provides a ready-to-use set of attributes with predescribed values, possible to be achieved by them (ISO 9126, Morisio and Torchiano, Torchiano and Jaccheri) [2, 16, 17]. Some of the selected frameworks also provide a precise number of descriptions, important for general framework development for COTS selection processes. The disadvantage is that most analyzed frameworks do not include soft factors during the selection process. It is worth to notice that only few selected frameworks provide an evaluation template for attributes with specified predescribed values. A financial aspect during the evaluation process is very often omitted (Table 2).

The comparative analysis of selected information tools allows to identify the general classification criteria [8, 9, 18–28] for a tool to a given category:

- agent-based approaches (INSEAS, RASCAL, ADIPS, goal-based workflow system for multiagent task coodrination),

Table 3 The relations between selected COTS information tools and the defined set of criteria

Name of criteria	Name of subcriteria	ADIPS	Agora	Cognitive Task Analysis	CompoNex Browsing	EKD	GAM	GBRAM	GBTCM	GOThIC	GQM	HAC	INSEAS	IPSCom	MDA	MoReCOTS	OntoManager	PLIB	RASCAL	SCB	SDM	SIBYL	SemaSC	Semantic Based Technique	SymOntoX
Additional improvements	Requirements analysis	-	-	yes	-	-	-	-	yes	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Modeling	-	-	-	-	-	-	yes	-	-	-	-	-	-	yes	-	-	-	-	-	yes	-	-	-	yes
	Acceptance of user's preferences	-	-	-	-	-	-	-	yes	-	-	-	-	-	-	-	yes	-	yes	-	-	yes	-	yes	-
	Using of portals	-	yes	-	-	yes	-	-	-	-	-	-	-	yes	-	yes	-	-	-	yes	-	-	yes	-	-
Classification citeria	Ontologies	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	yes	yes	-	-	-	-	-	-	yes
	Goal-oriented approach	-	-	yes	yes	yes	yes	yes	yes	yes	yes	-	-	-	-	-	-	-	-	-	yes	yes	-	-	-
	Semantic techniques	-	-	-	-	-	-	-	-	-	-	yes	-	-	-	-	-	-	-	-	-	-	yes	yes	-
	Web technology support	-	yes	-	-	-	-	-	-	-	-	-	-	yes	yes	yes	-	-	-	yes	-	-	-	-	-
	Using of agents	yes	-	-	-	-	-	-	-	-	-	-	yes	-	-	-	-	-	yes	-	-	-	-	-	-
Components searching method	Metasearching	-	-	-	-	-	-	-	-	-	-	-	-	-	-	yes	-	-	-	-	-	-	-	-	yes
	Searching based on categories	-	-	-	-	-	yes	-	-	-	yes	yes	-	-	-	-	-	-	-	-	-	-	-	-	-
	Searching based on key words	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	yes	-	-	-	-	-
	Searching based on key words and categories	-	yes	-	-	-	-	-	-	yes	-	-	-	yes	-	-	-	-	-	-	-	-	-	-	-
	Searching components from the Web	-	-	-	-	-	-	-	-	yes	-	-	-	-	-	yes	-	-	-	yes	-	-	-	yes	-
Phase of develop ment	Project phase	-	-	-	-	-	-	-	-	-	-	-	-	yes	-	-	-	-	-	-	-	-	-	-	-
	Ready-to-use solution	-	-	-	-	-	yes	-	-	-	yes	-	-	-	-	-	yes	-	-	-	-	-	-	-	-
	Prototype	-	yes	-	-	-	-	-	-	-	-	-	-	-	-	yes	-	-	-	yes	-	-	yes	-	-
Semantic technologies	Taxonomy	-	-	-	-	-	-	yes	-	yes	-	-	-	-	-	yes	-	-	-	-	-	-	yes	-	-
	Ontology improvement	-	-	-	-	-	-	-	yes	-	-	-	-	-	-	-	-	yes	-	-	-	-	-	-	yes
	Using of ontology manager	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	yes	yes	-	-	-	-	-	-	-
	Using of repository	yes	-	-	-	-	-	yes	yes	-	-	-	yes	yes	-	yes	-	-	yes	-	-	-	-	-	-
	Using of generic ontology	-	-	-	-	-	-	-	-	-	-	-	-	yes	-	-	yes	yes	-	-	-	-	-	-	yes
	Using of recommended system	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	yes	-	-	-	-	-	-
Supporting process of COTS components selection	Automatic components indexation	-	yes	-	-	-	-	-	-	-	-	-	-	yes	-	yes	-	-	-	yes	-	-	yes	-	-
	Information about components	-	-	yes	-	-	-	-	-	yes	-	-	-	-	-	-	-	-	-	yes	yes	-	yes	-	-
	Components tests	-	-	yes	yes	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Components location	-	yes	-	-	-	-	-	-	-	-	-	-	-	-	yes	-	-	yes	-	-	-	yes	-	-
	Searching/retrieval of components	-	yes	-	-	-	-	-	yes	yes	-	-	yes	-	-	yes	-	-	-	-	-	-	yes	-	-
Advanced searching mechanisms	Clustering multi-attribute data	-	-	-	-	-	-	-	-	-	-	yes	-	-	-	-	-	-	-	-	-	-	-	-	-
	Other improved mechanisms	-	yes	-	-	-	-	-	-	-	-	yes	-	yes	-	yes	-	-	-	yes	-	-	yes	-	-
	Using of metrics	-	-	-	-	-	yes	-	-	-	yes	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Natural language queries	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	yes	-

- sematic techniques (Sema-SC, semantic-based technique, hierarchical agglomerative clustering—HAC),
- classic tools supported by Web technology (Agora, SCB, MoreCOTS, IPSCom, MDA),

- goal-oriented approaches (GOThIC, cognitive task analysis, ISAC, EKD, SYBIL, SDM, REMAP, KAOS, GQM, GAM, GBTCM, GBRAM, GSN, NFR, reasoning loop model, CompoNex Browsing), and
- ontology-based approaches (OntoManager, PLIB, SymOntoX).

Due to limitations of space in this paper it is not possible to provide a comparative analysis of the information tools.

Based on the COTS frameworks and information tools characteristics the set of criteria was defined. On the basis of this analysis, 4 main criteria and 22 sub-criteria were defined for frameworks (Table 2) and 7 main criteria and 32 sub-criteria were defined for information tools (Table 3). In many cases the names of criteria should be generalized; it helps in limitation of the total number of criteria in the ontology project. Then the reasoning mechanism computes the results more efficiently and faster.

3 Knowledge-Based Approach to COTS Frameworks and Information Tools Ontology Construction

A knowledge-based approach to COTS frameworks ontology and COTS information tools ontology construction is proposed. In this case it is necessary to limit a domain of interest to the COTS framework and information tool domains. It is assumed that the ontology that supports COTS frameworks and COTS information tools selection should ensure freedom in requirements definition processes. Moreover, it should provide the knowledge systematization of COTS framework and information tool domains, and enable a time reduction for COTS component selection processes.

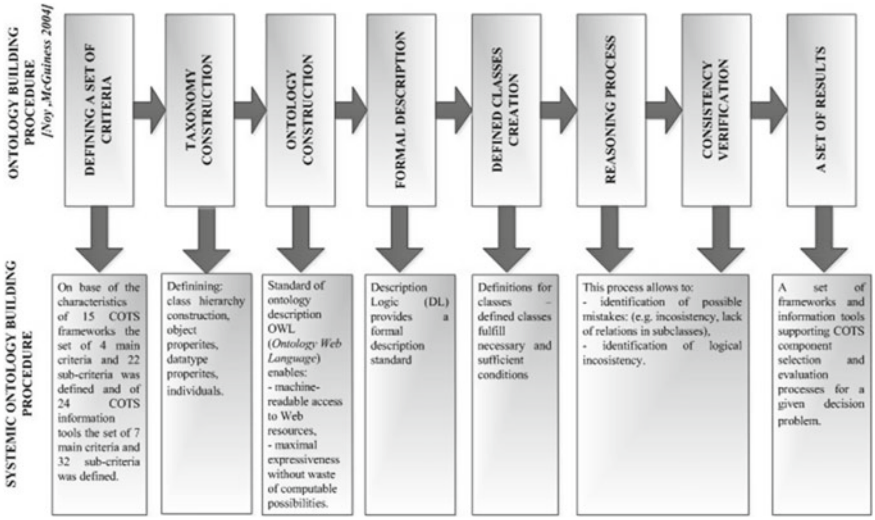
Decision makers may have different views of the same decision problem and they may look for different frameworks and information tools supporting COTS component selection processes. Moreover, decision makers do not have a broad knowledge of available COTS components on the market. The proposed ontologies help them to find a solution that suits their preferences the best. Decision makers define a set of preferences the solution should have. A reasoning mechanism computes the definitions and provides a set of results (a set of COTS frameworks and a set of COTS information tools) which fulfills predefined requirements. On the output decision makers obtain a set of selected frameworks and a set of selected information tools that help in COTS component selection processes. The higher the level of specification, the smaller the number of results [30].

3.1 Systemic Procedure of COTS Ontology Construction

The whole procedure is exactly the same for both the presented ontologies. The basis for the ontology construction was a thorough analysis of the considered solutions and then the experiment of identification of the set of criteria and sub-criteria. On this basis, the taxonomy was built. For both the presented ontologies supporting COTS component selection and evaluation, the set of criteria was created on the basis of available characteristics of these frameworks (Table 2) and information tools (Table 3). The taxonomy is separated for each analyzed COTS domain of interest (frameworks and information tools). The aim of the taxonomy is to ensure systematization and classification for particular solutions [31]. The definitions for classes were created with necessary and sufficient conditions. It allows to select the most suitable solutions for a decision maker.

The schema presents a general procedure of ontology construction supporting COTS framework selection processes (Fig. 1). This procedure is divided into eight phases: (1) defining a set of criteria, (2) taxonomy construction, (3) ontology construction, (4) formal description, (5) defined classes creation, (6) reasoning process, (7) consistency verification, (8) a set of results. A domain of modeling encompasses a set of 15 COTS frameworks and 24 COTS information tools. The general aim of the proposed systemic procedure is to provide the knowledge of methodological aspects and to ensure a specified guideline supporting COTS component classification and selection processes.

The two separated ontologies were built using the Protégé application (frameworks: http://www.semanticweb.org/ontologies/2013/Ontology1293721946681.owl and information tools: http://www.semanticweb.org/ontologies/2013/Ontology1293661088042.owl). The applied technology standard is OWL (Ontology Web Language).



On base of the characteristics of 15 COTS frameworks the set of 4 main criteria and 22 sub-criteria was defined and of 24 COTS information tools the set of 7 main criteria and 32 sub-criteria was defined.

Definining: class hierarchy construction, object properites, datatype properites, individuals.

Standard of ontology description OWL (*Ontology Web Language*) enables:
- machine-readable access to Web resources,
- maximal expressiveness without waste of computable possibilities.

Description Logic (DL) provides a formal description standard

Definitions for classes – defined classes fulfill necessary and sufficient conditions

This process allows to:
- identification of possible mistakes: (e.g. incosistency, lack of relations in subclasses),
- identification of logical incosistency.

A set of frameworks and information tools supporting COTS component selection and evaluation processes for a given decision problem.

Fig. 1 The phases of COTS frameworks and information tools ontology construction

4 Case Studies: Ontology Supporting COTS Component Selection Processes

The case study presents practical examples of COTS framework and information tool ontology applications. It is divided into two parts: first, a practical usage of COTS framework ontology is provided. The second part encompasses a practical example of COTS information tool ontology applications. Due to the limited space only a small part of COTS information tools ontology is presented in this case study.

4.1 Case Studies: Ontology Supporting COTS Component Selection Processes Considering a Proper COTS Framework Choice for a Given Decision Problem

It is supposed that a decision maker is looking for the COTS framework that fulfills a set of predefined requirements. The preferable framework should satisfy at least one of the following criteria: (1) Evaluation of attributes, or (2) Evaluation of financial aspects, or (3) Requirements analysis, or (4) Process systematization, or (5) User attendance in selection process. The application of the reasoning mechanism provides a set of results with regard to the predefined requirements. In this case nine frameworks (CRE, CBCPS, OTSO, PORE, CEP, CAP, STACE, Carney and Wallnau, Morisio, and Torchiano) fulfill at least one of a defined set of criteria (Fig. 2).

Next, a decision maker changes a set of criteria to the following: (1) Process recurrence, (2) Process systematization, (3) Evaluation of product. It is obligatory for a preferred COTS framework to fulfill all of these defined requirements. The reasoning

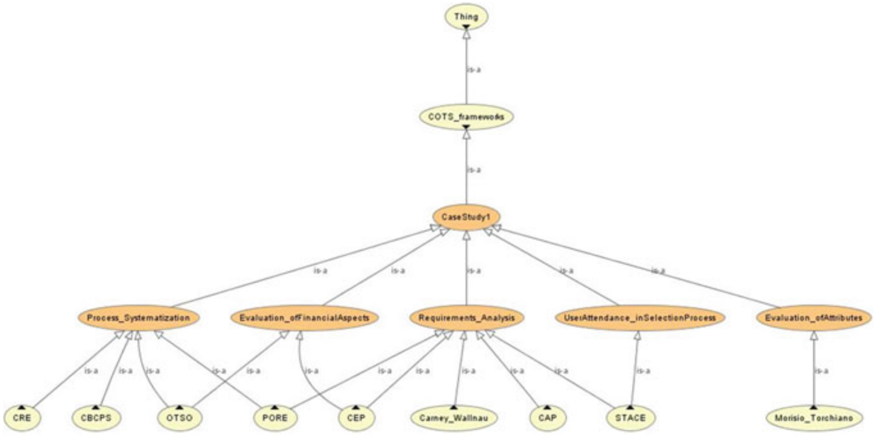


Fig. 2 A practical example of COTS ontology application—a limited set of results

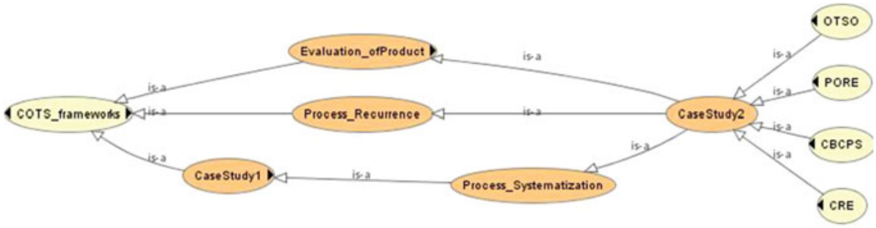


Fig. 3 A practical example of COTS ontology application—a limited set of results

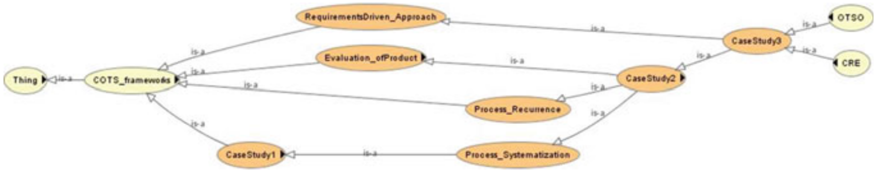


Fig. 4 A practical example of COTS ontology application—a limited set of results

mechanism provides the more precise set of results (Fig. 3). This set of criteria is satisfied by the following COTS frameworks: OTSO, PORE, CBCPS, and CRE.

Moreover, one criterion extra (Requirements analysis) was added. As a consequence, a new set of results (OTSO, CRE) was provided. It is worth to notice that the more valuable the requirements definition, the smaller the number of identified results (Fig. 4).

It is possible to present this query using DL (Description Logic) Query mechanisms. The same query was posed using a DL Query mechanism. The identified set of results is exactly the same as earlier. It is possible to specify a nonlimited set of queries for the COTS framework ontology. Moreover, the decision maker does not have to have a broad knowledge of frameworks, but can still make a reasonable choice.

4.2 Case Studies: Ontology Supporting COTS Component Selection Processes Considering a Proper COTS Information Tool Choice for a Given Decision Problem

The case study presents a practical example of COTS information tool ontology applications. It is supposed that a decision maker is looking for the COTS information tool that fulfills a set of predefined requirements: (1) is based on goal-oriented approach or (2) uses semantic techniques.

The application of the reasoning mechanism provides a set of results with regard to the predefined requirements. In this case, 13 information tools fulfill at least one of requirements: GOThIC, Cognitive Tasks Analysis, CompoNex Browsing, GBRAM,

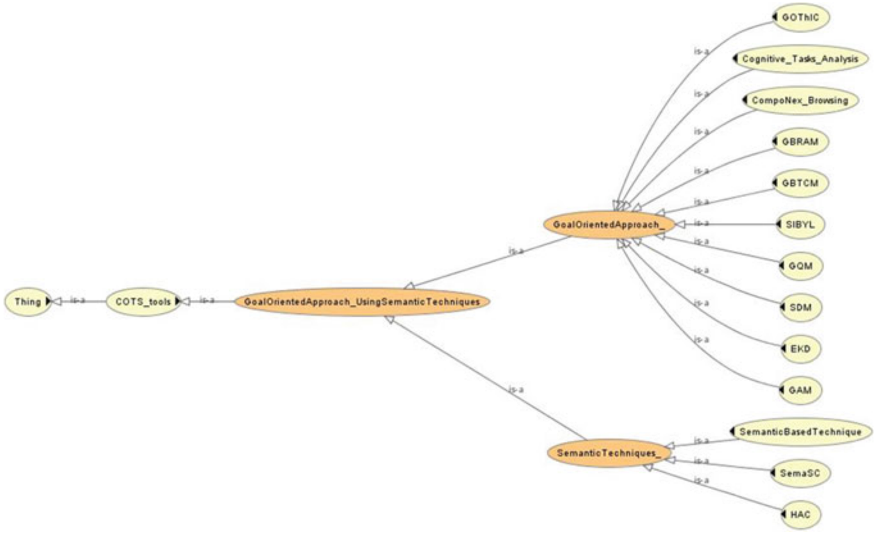


Fig. 5 A practical example for COTS ontology application—a limited set of results

GBTCM, SIBYL, GQM, SDM, EKD, GAM, Semantic-based technique, SemaSC and HAC (Fig. 5).

This case study presents only a small part of the practical application of both ontologies: COTS framework ontology and COTS information tool ontology (due to limited space in this paper). It is possible to specify a nonlimited set of queries for presented ontologies. Moreover, a decision maker decides about a level of specification—the higher the level of specification the smaller the number of results. It is worth to notice that domain ontology (domain of COTS framework or COTS information tool) allows to compute the results in shorter time than a global ontology for COTS.

5 Conclusion

This paper presents the knowledge-based approach to COTS framework and information tools selection processes. On this basis, OWL standard was used to create the ontologies: COTS frameworks ontology and COTS information tools ontology. The general aim of the COTS ontology construction was to provide a systematic and repeatable way for selection of a proper COTS framework for a given decision problem.

The COTS frameworks and information tools provide information about process classifications supporting COTS component selection, and they present issues for COTS component selection processes. The analysis of the literature allows to identify

15 COTS frameworks and 24 information tools. On a basis of the specified characteristics of these solutions the ontologies were built. Due to the limited space of publication, only a small portion of the practical application of a COTS framework and information tool ontologies were presented. It is worth to notice that it is possible to specify a nonlimited set of queries for the ontologies. Moreover, the decision maker does not have to have a broad knowledge of frameworks and information tools, but can still make a reasonable choice.

It is worth to emphasize that the knowledge-based approach to COTS software selection was composed of three separated ontologies: ontology for methods and techniques [12], ontology for information tools, ontology for frameworks (provided in this paper) and ontology for COTS ERP components. The general aim was to provide a complex process to support the process of COTS software selection. The ontologies for methods and techniques, information tools and frameworks provide methodological approaches to COTS software selection, whereas the ontology for COTS ERP components helps in component selection processes, and provides a specified knowledge of the available COTS ERP components.

Appendix 1

See Table 4.

Table 4 The characteristics of the selected COTS frameworks

Framework	Application areas	Evaluation process
SSEF (Software System Evaluation Framework), Boloix and Robillard [13]	Technology evaluation, product evaluation, and its influence on the organization; it indicates the strengths and weaknesses of the technology	It is based on organized attributes and divided into three dimensions: project, system and environment. It allows to accept only three values: basic, medium and advanced. Top-down approach; information is estimated from different points of view; several applications of this framework are available
Carney and Wallnau [15]	Software products evaluation with respect to the high level of details	It is based on four basic postulates that define COTS software evaluation. Three of them indicate a conceptual basis for COTS evaluation; the last of them indicate specification necessity of COTS software evaluation
OTSO (Off-The-Shelf-Option), Kontio et al. [1]	Simplification of systematic, repeatable and requirements-driven process of selection with respect to financial and quality aspects	It is based on task definitions in selection process using input and output criteria; the incremental, hierarchical and specific definitions of evaluated criteria

Table 4 (continued)

Framework	Application areas	Evaluation process
Delta technology framework [14]	Support in software technology evaluation process	It is based on checking software features with their equivalents; it provides a systematic approach based on modeling and experience. It is based on three identification phases and delta features estimation: modeling, experience project and experience evaluation
PORE (Procurement-Oriented Requirements Engineering), Maiden and Ncube [29]	Support in requirements acquisition for COTS selection process; iterative requirements acquisition process	It joints together requirements engineering methods and other techniques such as: features analysis and multi-criteria decision support. The whole process encompasses the following steps: requirements acquisition, describing and analysis of requirements and modeling and analysis of COTS candidates at the same time
CAP (COTS Acquisition Process), Ochs et al. [7]	COTS evaluation process should be fitted on base of available requirements for each project	Process pre-definition, short tasks, and heuristics description. Performance and effectiveness of the process should be ensured by heuristics methods. This method provides well-defined, systematic, and repeatable process
STACE (Social-Technical approach to COTS Evaluation), Kunda and Brooks [32]	Social-technical approach for evaluation of a set of criteria	It bases on social-technical approach in evaluation process; a user participates in evaluation process
CRE (COTS-Based Requirements Engineering method), Alves and Castro [4]	Simplification of systematic, repeatable, and requirements-driven process of COTS software selection	Nonfunctional requirements analysis in COTS selection and evaluation process. It is composed of four main phases: identification, description, evaluation, and validation. MCDA model is used to estimation of a software quality for a given project
CEP (Comparative Evaluation Process Activities), Cavanaugh and Polen [6]	It is based on postulate that the reliability increment of data source has a higher importance on the results reliability	It bases on calculate spreadsheet that supports a decision maker in COTS software comparison (based on differentiated criteria). The model construction bases on decision models theory. A result for each of the analyzed products allows to define if a product fits a given project

(continued)

Table 4 (continued)

Framework	Application areas	Evaluation process
FCS (Framework for COTS Selection), Wanayama et al. [33]	It supports the access to the expert's information and experience acquisition from previous evaluation processes	It is based on defining criteria template including weights using MCDA method; it is supported by DSS cooperating with FCS
CBCPS (Contract-Based COTS Product Selection), Ye and Kelly [34]	It provides additional evaluation in searching process; systematic and repeatable process	General aim of CBCPS is to create untended relations between software and critical safety of a system. It enables a systematic, repeatable, and risk-driven software selection process. It supports an integration process of COTS products and ensures a precise critical safety analysis of a system. CBCPS allows to indicate a set of evaluation criteria ob base of safety requirements
Morisio and Torchiano [2]	The organization of available attributes in consistent way; a proposal of the new attributes	A number of attributes was presented with possible specified values characterizing COTS products. COTS product is described by a single value of each attribute—it is possible to prescribe only one value. Different products with the same set of values belong to the same class
Framework ISO 9126 [16]	ISO/IEC standard 9126 provides the set of attributes for software quality evaluation	It is based on a specified definition of each of attributes and it presents a possible way of measure of these attributes. It is composed of six attributes: functionality, reliability, usability, performance, modifiability, portability
Torchiano and Jaccheri [17]	The identification of attributes and assignment of the values scale	It is based on defining attributes to use them for particular situations supporting COTS components selection, technology or tools
Carney and Long [5]	It is based on COTS product classification with two-dimension Cartesian space; dimensions are defined by origin and modifiability	A dimension of origin directs on the software production process. A modifiability dimension defines the possible or necessary modifications by system developer who uses a particular component

References

1. Kontio, J., Chen, S., Limperos, K., Tesoriero, R., Caldiera, G., Deutsch, M.: A COTS selection method and experiences of its use. Twentieth Annual Software Engineering Workshop, Greenbelt (1995)
2. Morisio, M., Torchiano, M.: Definition and classification of COTS: a proposal. Accepted at ICCBSS, Orlando (FL), 4–6 February 2002
3. Jelokhani-Niaraki, M., Malczewski, J.: A web 3.0-driven collaborative multicriteria spatial decision support system. Systèmes, Modélisation, Géostatistiques 620–630 (2012)
4. Alves, C., Castro, J.: CRE: A systematic method for COTS components selection. In: XV Brazilian Symposium on Software Engineering (SBES), Rio de Janeiro, Brazil (2001)
5. Carney, D., Long, F.: What do you mean by COTS? IEEE Softw. 83–86 (2000)
6. Cavanaugh, B.P., Polen, S.M.: Add decision analysis to your COTS selection process, J. Def. Softw. Eng. (2002)
7. Ochs, M.A., Pfahl, D., Chrobok-Diening, G.: A method for efficient measurement-based COTS assessment and selection - method description and evaluation results. In: IEEE 7th International Software Metrics Symposium, pp. 285–296. England, London (2001)
8. Ayala, C., Franch, X.: A goal-oriented strategy for supporting commercial Off-The-Shelf components selection. In: Proceedings of the 9th International Conference on Software Reuse (ICSR), Torino, Italy. Lecture Notes in Computer Science, vol. 4039, pp. 1–15 (2006)
9. Mohamed, A., Ruhe, G., Eberlein, A.: COTS selection: past, present and future. In: Proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'07) (2007)
10. Tarawneh, F., Baharom, F., Yahaya, J., Ahmad, F.: Evaluation and selection COTS software process: the state of the art. The society of digital information and wireless communications (SDIWC). Int. J. New Comput. Archit. Appl. (IJNCAA) 01/2011 **2**, 344–357 (2011)
11. Wanyama, T., Far, B.: An empirical study to compare three methods for selecting COTS software components. Int. J. Comput. ICT Res. **2**, 34 (2008)
12. Konys, A., Wątróbski, J., Różewski, P.: Approach to practical ontology design for supporting COTS component selection processes, ACIIDS 2013. In: Selamat, A., et al. (eds.) ACIIDS 2013, Part II. LNAI, vol. 7803, pp. 245–255. Springer, Heidelberg (2013)
13. Boloix, G., Robillard, P.: A software system evaluation framework. IEEE Comput. **12**(8), 17–26 (1995)
14. Brown, A.W., Wallnau, K.C.: A framework for systematic evaluation of software technologies. IEEE Softw. **13**(5), 39–49 (1996)
15. Carney, D.J., Wallnau, K.C.: A basis for evaluation of commercial software. Inf. Softw. Tech. **40**, 851–860 (1998)
16. ISO: Information Technology - software product evaluation - quality characteristics and guidelines for their use, Int. Standard ISO/IEC 9126, ISO (1991)
17. Jaccheri, L., Torchiano, M.: Classifying COTS products. In: European Conference on Software Quality, Helsinki (2002)
18. Aguirre, J.: IPSComp: Intelligent Portal for Searching Components. In: Vrije Universiteit Brussel - Belgium Faculty of Sciences in Collaboration with Ecole des Mines de Nantes - France (2005)
19. Bourlard, H., Konig, Y., Morgan, N.: REMAP: Recursive estimation and maximization of a posteriori probabilities application to transition-based connectionist speech recognition. In: International Computer Science Institute (ICSI), Berkeley, California EECS Department, University of California, Berkeley, California TR-94-064, March (1995)
20. Cyra, Ł., Górski, J.: Extending GQM by argument structures. In: Lecture Notes in Computer Science, vol. 5082. Berlin/Heidelberg (2008)
21. Kiemen, M.: A triple loop model of agent cognition (ECCO). Vrije Universiteit Brussel, Belgium (2003)

22. Kotonya, G., Hutchinson, J.: A service-oriented approach for specifying component-based systems. In: Proceedings of the 6th International Conference Software Systems (ICCBSS), pp. 150–162 (2007)
23. Lee, J.: Sibyl: A Tool for Managing Group Decision Rationale. Technical Report, Massachusetts Institute of Technology Cambridge (1990)
24. Morisio, M., Seaman, C.B., Basili, V.R., Parra, A.T., Kraft, S.E., Condon, S.E.: COTSbased software development: processes and open issues. J. Syst. Softw. **61**(3), 189–199 (2002)
25. Simmons, G.L., Dillon, T.S.: Towards an ontology for open source software development. In: Damiani, E., Fitzgeralg, B., Scacchi, W., Scotto, M., Succi, G. (eds.) Open Source Systems, IFIP International Federation for Information Processing, vol. 203, pp. 65–75. Springer, Boston (2003)
26. Tonu, S.A., Tahvildari, L., Tahvildari, S.: A framework to incorporate non functional requirements into UML models. In: TahvildariSoftware Technologies Applied Research LabSoftware LabDepartment of Electrical & Computer Engineering (November 2008)
27. Vitharana, P., Zahedi, F., Jain, H.: Knowledge-based repository scheme for storing and retrieving business components: a theoretical design and empirical analysis. IEEE Trans. Softw. Eng. **29**(7), 649–664 (2003)
28. Wanyama, T., Far, B.H.: Repositories for COTS Selection. In: Proceedings of the Canadian Conference on Electrical and Computer Engineering (CCECE'06), pp. 2416–2419 (2006)
29. Maiden, C.N., Ncube, A.M.: PORE: procurement-oriented requirements engineering method for the component-based systems engineering development paradigm. In: 2nd International Workshop on Component-Based Software Engineering, Los Angeles (1998)
30. Konys, A., Wątróbski, J.: A model of ontology supporting COTS component selection process in management information system domain. In: Advanced Information Technologies for Management AITM'2010. Wroclaw University of Economics Research Papers (2010)
31. Konys, A.: Ontologies supporting COTS software components selection and evaluation. In: Advanced Information Technologies for Management AITM'2011. Wroclaw University of Economics Research Papers (2011)
32. Kunda, L., Brooks, D.: Applying Social-Technical Approach for CotsSelection. In: Proceeding of 4th UKAIS Conference, University of York (1999)
33. Wanyama, T., Far, B.H.: A Multi-Agent Framework for ConflictAnalysis and Negotiation: Case of COTS Selection, Transactions of theInstitute of Electronics, Information and Communication Engineers:Special Issue on Software Agent and its Applications–vol. E88-D, No.9,pp. 2047–2058 (2005)
34. Ye, F., Kelly, T.: COTS Product Selection for Safety-Critical Systems, Lecture Notes in Computer Science. vol. 2959, pp.53–64 (2004)

The Investment Strategy Based on the Difference of Moving Averages with Parameters Adapted by Machine Learning

Antoni Wiliński and Michał Zabłocki

Abstract In this paper, the authors present an investment strategy based on moving averages (MA). The strategy on the basis of the relationship between two moving averages classifies events of opening long and short position. To gain desirable results it was enriched by extra filtering mechanisms, such as: StopLoss, first derivative of the difference of moving averages, and additional buffers within the classification rules—these parameters constitute the parameters space. The whole concept is based on machine learning principles. According to these principles, the values of the parameters are computed during the learning phase and applied in simulated trading during the testing phase. The experiments conducted showed that the strategy is effective for EURUSD 1 h currency pair.

Keywords Machine learning · Moving average · Optimization · Stock market

1 Introduction

The approach to forecast the future stock market prices presented in this paper is based on techniques using past prices. Two moving averages of prices are used to generate two kinds of signals—recommendations to open a long or a short position. This should be considered as a process of classification based on the relationship between those moving averages. Next, it was necessary to enrich the approach by some filter rules, which resulted in obtaining some simple investment strategies. In the history of global trading markets, such strategies (based on moving averages) belong to the simplest and certainly the first concepts of searching efficient models

A. Wiliński · M. Zabłocki (✉)
Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, Zołnierska 49, 71-210 Szczecin, Poland
e-mail: awilin@o2.pl

M. Zabłocki
e-mail: mzablocki@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_18

for forecasting or classification [1–5]. The criticism of their low effectiveness is commonly known [4–9].

Generally one can agree with these conclusions, however, enriching the strategies based on moving averages by new filters and classification rules can lead to surprisingly positive results [10, 11]. Of course, there may exist some more effective strategies not initiated by the moving averages, or ignoring them entirely [7, 8, 12–14].

In this paper, the concepts of certain transactional algorithms are presented. These algorithms are initially based on moving averages and are further successively developed. The following algorithms are described here:

- MA1—the basic algorithm—the examination of the general concept of opening positions "inwards" (explained later); this strategy brings poor results after taking into account the costs of the transactions; it was an incentive to initiate research on the difference of moving averages;
- MA2—improvement of the opening formula on the basis of the first derivative of the moving averages difference—dMA;
- MA3—testing of the variability of the optimal parameters' values in time;
- MA4—application of machine learning—using the best parameters from preceding learning phase to perform tests in new data window (for data not used during establishing the optimal parameters values).

The strategy is as simple as possible, but not simpler than it can be. It is both simple and effective. Meeting these expectations—of simplicity and effectiveness—is a great challenge. The solution is based on machine learning principles [12, 15] of continuous correction of parameters of the strategy and of application of the strategy in the testing phase on the data not used before.

2 MA1—The Basis of the Investment Strategy

The strategy is based on the simplest one of the possible simple rules—opening the position at the beginning of the candle, according to the relationship between two moving averages. For the purpose of this article, a Moving Average (MA_n) is an average of closures of the last few, several, or even tens of candles. In MA_n, n is the number of previous candles taken into account in the calculations of the average value.

$$MA_n(i) = mean(Close(i - n + 1) : Close(i - 1)), \tag{1}$$

for i = 1, 2, ..., I, where I—size of the time window (number of candles), in which the simulation is performed.

The $MA_n(i)$ variable should be read as the average value of closing prices of candles from $Close(i - n + 1)$ to $Close(i - 1)$. Index i will denote the current candle. Thus, let $Close(i - 1)$ represent the closing price of the candle preceding

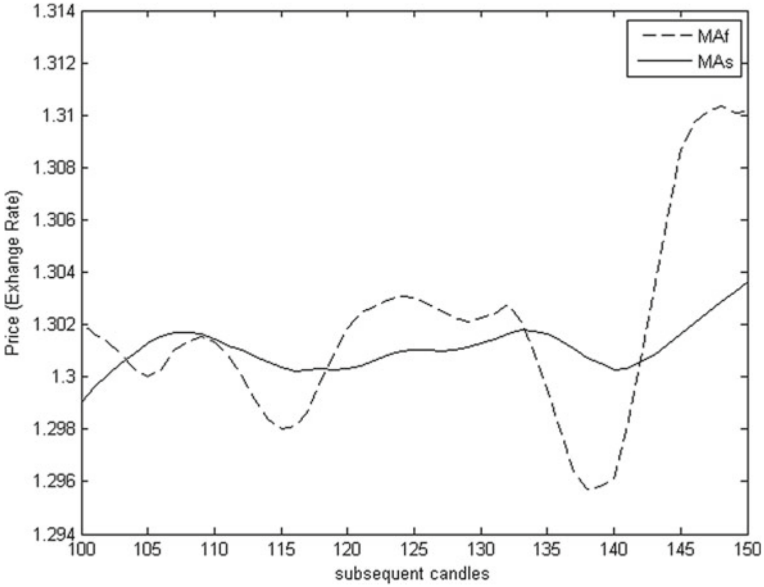


Fig. 1 An example of “slow” and “fast” MA trajectory for s = 20 and f = 5 for the fragment of EURUSD 1 h time series

the current one (on the Forex market it is practically the same as the opening price of the current candle).

The fundamental concept of the strategy comes down to calculating two moving averages MA_s (slow moving average) and MA_f (fast moving average), where $s > f$, and checking them for intersections (for the change in the sign of the averages' difference). The adjectives “slow” and “fast” are associated with the symbolic switch to the frequency domain—the “slow” average is a slowly varying one, of the lower frequency. The investor will perceive a similar impression when comparing those two curves on a graph (see Fig. 1).

The MA1 algorithm is based on a simple decision rule, commonly used in many sources [6, 12]. If intersection of slow and fast averages occurs, then it is a signal for position opening [6, 9, 10]. The position is opened according to the “inwards” rule:

$$IF\ MA_f(i) > MA_s(i)\ \ THEN\ short \tag{2}$$

$$IF\ MA_f(i) < MA_s(i)\ \ THEN\ long \tag{3}$$

for i = 1, 2, . . . , I.

In (2) and (3) the keywords long and short represent an order to open an appropriate position. The issue, when this position should be closed, is a separate problem, not considered at this point. Figure 2 depicts the idea of opening the position in the points of averages intersection.

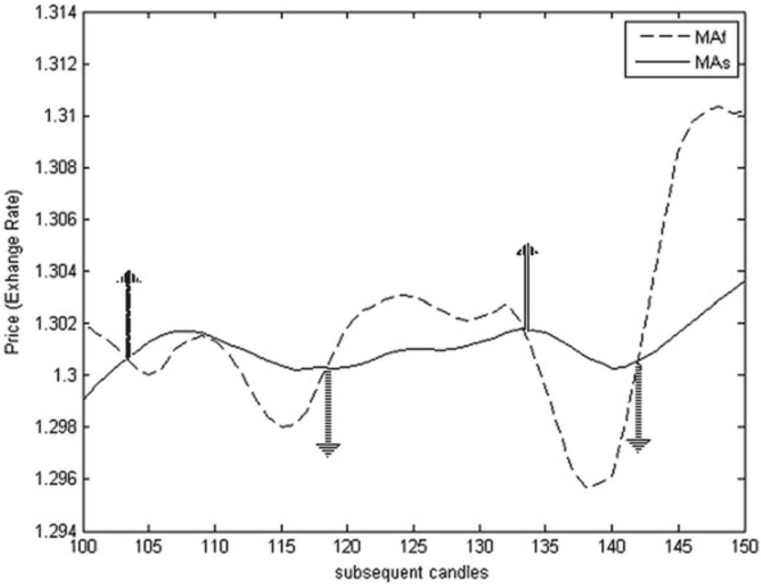


Fig. 2 An example of decision-making situations at the intersection points of two moving averages. The chart depicts the rule of opening positions "inwards." The first recommendation is to open the long position

The rule mentioned before, of opening positions "inwards", represents the adoption of the following hypothesis: if the fast moving average crosses the slower one from the bottom up, then it will probably revert and come back soon. Consequently, the prices are expected to decline, so it is rational to open a short position. Conversely, if the fast average crosses the slower one from the top, then the opposite behavior should be recommended—to open a long position. The observation of the graph (preferably in the machine learning mode) should allow to infer the decision, as to when (after how many candles) it would be best to close the opened positions.

The strategy includes an important variable called spread—it is a symbol of transactional costs charged by an Internet broker. Initially, those costs were assumed to be zero, and the subsequent study would reveal their destructive force.

In the next step, the parameters space was expanded by another three parameters. Two of them are additional buffers, which are taken into account in position opening rules (2) and (3). In order to convert these rules, an additional variable was introduced—the difference of moving averages *dMA*:

$$dMA(i) = MAf(i) - MAs(i) \tag{4}$$

for i = 1, 2, . . . , I.

After adoption of the two additional parameters, the long and short positions opening rules take the following form:

$$IF\ dMA(i) > bd\ \ THEN\ short \tag{5}$$

$$IF\ dMA(i) < -bu\ \ THEN\ long \tag{6}$$

for i = 1, 2, ..., I and where bu, bd are the two additional parameters, the values of which are subject to optimization.

The last of the five considered parameters is SL—Stop Loss. It is a widely used in automated trading variable representing investor's aversion to loss—SL is the highest accepted value of loss. This variable is subject to optimization in the proposed algorithm.

Considering the position closings caused by an unacceptable, but local loss, a full information contained in the so-called OHLC candle was taken into account. For example, if it was assumed that the positions should be opened according to the rules introduced in (5) and (6) and closed at the end of the current candle, then the potential profit can be calculated as follows:

- in the case of meeting the long position opening conditions:

$$z(i) = C(i) - C(i-1) - spread \tag{7}$$

  for $dMA(i) < -bu$ and for i = 1, 2, ..., I, where $C(i)$ denotes the closing value of the ith candle;
- in the case of meeting the short position opening conditions:

$$z(i) = C(i-1) - C(i) - spread \tag{8}$$

  for $dMA(i) > bd$ and for i = 1, 2, ..., I.

If after the position is opened the adverse course change occurs inside the considered candle, causing accumulation of loss, then the SL mechanism will be enabled in the following way:

- in the case when the long position was opened:

$$z(i) = -SL - spread \tag{9}$$

  when $C(i-1) - L(i) > SL$, for i = 1, 2, ..., I, where $L(i)$ is the smallest value of price within the ith candle.

It should be interpreted as exceeding the acceptable level of difference between the level of a long position opening $C(i-1)$ at preceding $(i-1)$ candle closing, and the lowest value $L(i)$ within the ith candle.

Similarly, in the case when the short position was opened:

$$z(i) = -SL - spread \tag{10}$$

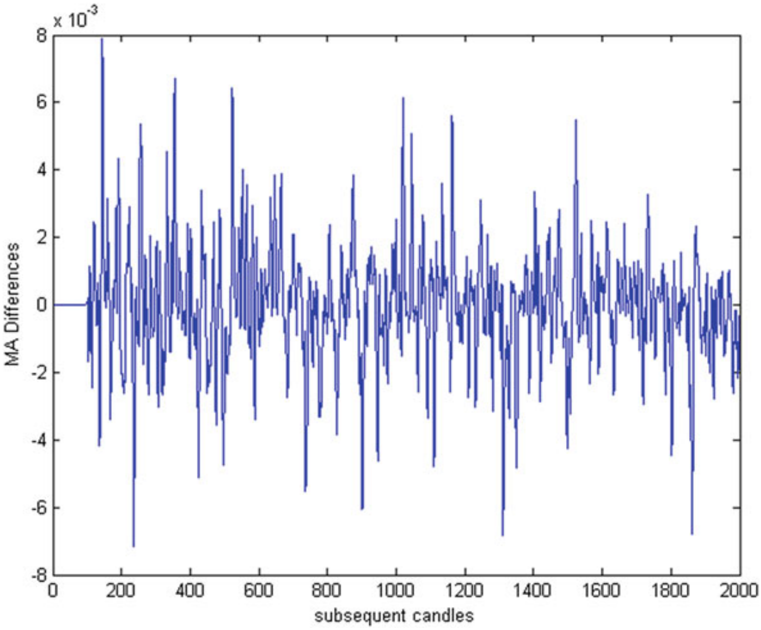


Fig. 3 A plot illustrating the variability of the difference of moving averages for large time series fragment (2,000 candles)

when $-C(i-1)+H(i) > SL$, for i = 1, 2, ..., I, where $H(i)$ is the largest value within the currently analyzed candle.

The accumulation of capital (cumulative profit) is expressed as follows:

$$Zs(i) = Zs(i-1) + z(i) \tag{11}$$

where $Zs(i)$ is the cumulative profit after the ith candle, z(i) is a profit in the current candle.

Below the following figures are presented:

- $Z_r ec$—is the plot of cumulative profit curve $Zs(i)$ for the test case, when the highest overall profit was achieved;
- *dMA*—is a plot of moving averages difference variability;
- *dMA*$(t1 : t2)$—is an arbitrarily selected subsequence of *dMA* from the point t_1 to t_2 in order to illustrate the idea of utilizing *dMA* variability in further improvements of the algorithm.

After running the simulations for MA1, the following results were obtained (see Figs. 3, 4 and 5).

It is worth noting that the difference of moving averages oscillates between positive and negative values in a rather regular way. The extremes of *dMA* are observable

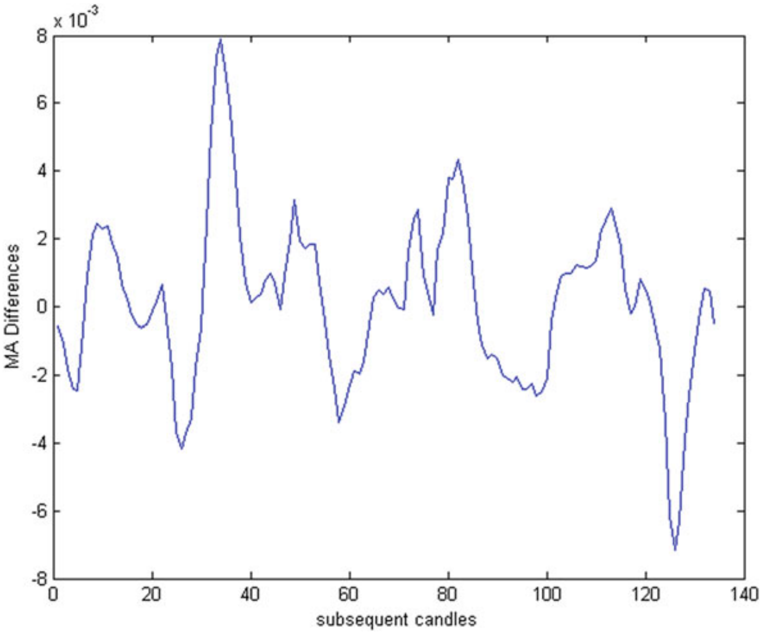


Fig. 4 A fragment of *dMA* illustrating the justification of searching the zero points and building the strategy based on intersections of averages

and empirically identifiable. In Fig. 4 one can observe the regular switching from positive to negative and vice versa.

The regularity in *dMA* in Fig. 3 allows to formulate a hypothesis (the strategy) to establish the optimal bd and bu buffer values, and after exceeding them to open the positions, which would be closed after one candle. Whether such a strategy is appropriate, was verified by the curve of cumulative profit—see Fig. 5.

The roots of *dMA* function also have specific intervals (see Fig. 4). Their frequency is not changing rapidly—of course for the chosen averages (fast and slow). Observing the variability in this figure led to formulating a hypothesis, that closing the positions after one candle is not the best solution. Perhaps the positions should be closed after k candles, where the optimal value of k would be searched for.

Figure 5 presents a surprisingly good answer to the question of whether the described algorithm is effective in predictive terms. The plot shows a regular growth of capital (except for the first segment), which confirms the algorithm's predictive abilities (there are possibilities of finding the optimal parameters values—learning them) in quite a large time horizon of 2,000 hourly candles. However, probably every good predictive algorithm [6] can be spoiled by corresponding transactional costs. Thus, the algorithm was tested for the market value of spread. After setting spread value to 0.00016, the cumulative profit as in Fig. 6 was obtained.

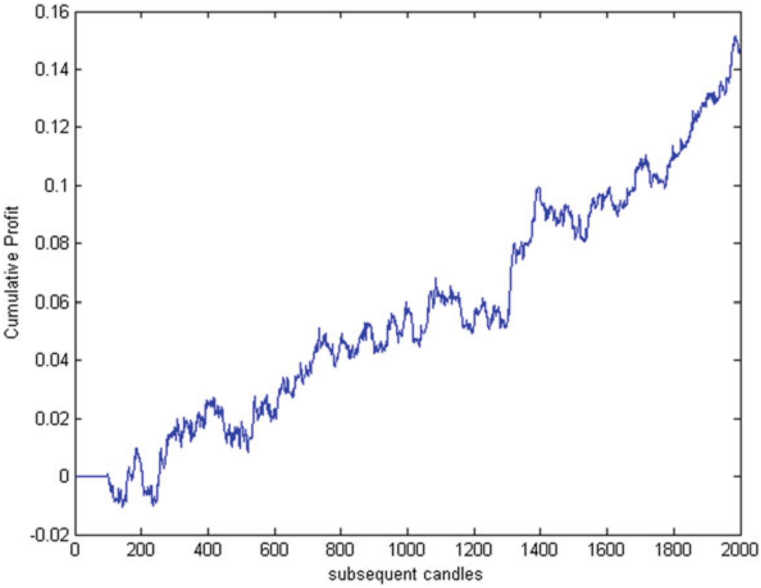


Fig. 5 A plot of capital accumulation for zero value of spread variable

The value of spread (1, 6 pips) was taken from one of the popular brokerage platforms1, of course for the EURUSD currency pair considered in this paper.

3 MA2—Improvement of the Opening Formula

This annoying result compels us to refrain from the application of this algorithm and to search for better solutions in terms of transaction volume. The inspiration for further research became a graph in Fig. 4. The plot shows that it makes no sense to open a position (e.g., a long position) each time *dMA* exceeds a certain buffer (in the case of long positions—bu buffer). Very important will be an answer to the question, whether at the moment of fulfilling this condition, the *dMA* curve rises, or falls. Let us call this feature the first derivative of *dMA*, defined as:

$$dMA'(i) = dMA(i) - dMA(i-1) \tag{12}$$

In order to open a long position it is expected that, besides meeting the previous condition (6), the value of *dMA* derivative, as specified by (12), will be positive.

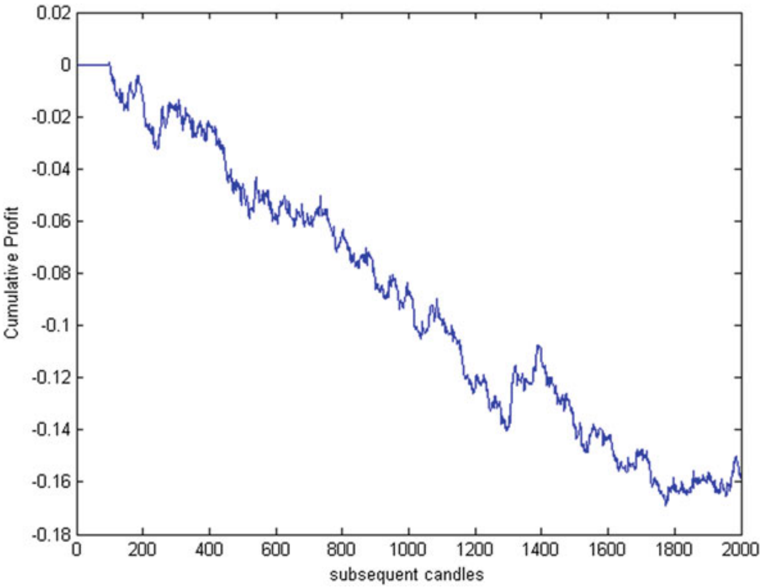


Fig. 6 A plot of capital accumulation after introducing the transactional costs in MA1 algorithm

Similarly, to the condition of opening a short position (5), a condition of negative value of dMA derivative (12) should be appended.

$$IF\ dMA(i) > bd\ \ AND\ \ dMA'(i) < 0\ \ THEN\ short \tag{13}$$

$$IF\ \ dMA(i) < -bu\ \ AND\ \ dMA'(i) > 0\ \ THEN\ long \tag{14}$$

for i = 1, 2, …, I.

The MA2 algorithm was launched with similar data as the MA1 algorithm, with the spread value of 1, 6 pips (such as in the Fig. 6). Obviously, the results were depressing and forcing to searching for additional conditions. Of course the simplest solution would be to end the investigation and to stop the development of the strategy after observing the plot in Fig. 6.

However, after adding the first derivative of *dMA*, taking into account the additional parameter of closing the position after k candles and expanding the research horizon up to 5,000 candles, the results were obtained as shown by a cumulative profit trajectory in Fig. 7.

The cumulative profit in the horizon of 5,000 candles for constant, best values of parameters is the result of long term learning, searching for the optimal parameter

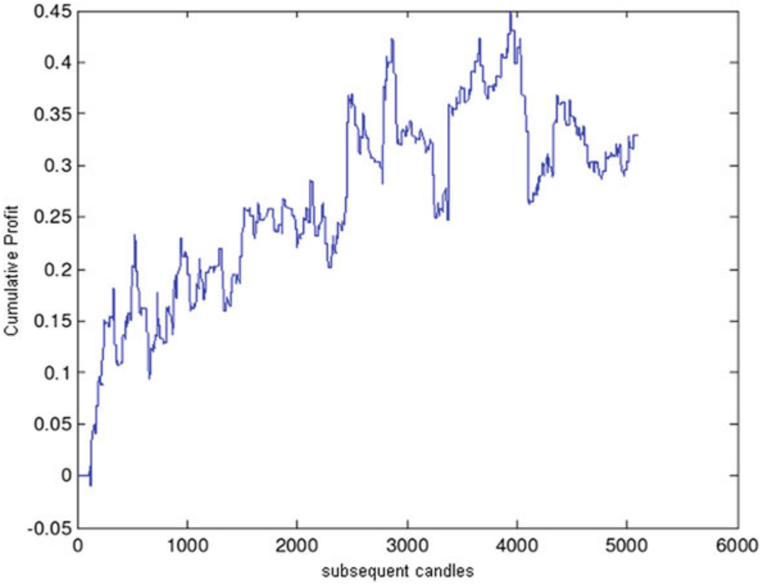


Fig. 7 The cumulative profit for the MA2 algorithm, 5,000 candles and initially very limited ranges of parameters variability

values for such a long time period. A quite obvious quality criterion for the strategy was introduced—the result at the end of simulation period Zsk (the higher the better, the higher the profit). Figure 7 shows the cumulative profit trajectory for the best selected parameters.

Every final result depends on the constant values of six parameters (p_1, p_2, ..., p_6). Each of these parameters takes discrete values from a certain empirically established set:

$$p_i \in p_{i_1}, p_{i_2}, \ldots, p_{iK_i}; \tag{15}$$

for i = $1, 2, \ldots, 6$, where $K_1, K_2, \ldots, K_6$ is a set of natural numbers defining the cardinality of the sets for each parameter values.

Each parameter is being changed K_i times, e.g. p_1 takes values $p_{1_1}, p_{1_2}, \ldots, p_{1_{K_1}}$. That way, the space of possible parameter values is formed by a hyperspace grid with $K_1 \times K_2 \times \cdots \times K_6$ nodes.

Each index $k_i = k_1, k_2, \ldots, K_i$ corresponds to a certain real value. The parameters values for presented data and variability ranges are calculated as follows:

$$\begin{aligned}
&\text{Parameter 1: } p_1\\
&\text{The real value of } p_1 \text{ is denoted as } s = k_1;\\
&\text{Considered variability range of parameter } p_1 : k_1 = 1 : 3\\[1em]
&\text{Parameter 2: } p_2\\
&\text{The real value of } p_2 \text{ is denoted as } w = s + k_2;\\
&\text{Considered variability range of parameter } p_2 : k_2 = 1 : 3\\[1em]
&\text{Parameter 3: } p_3\\
&\text{The real value of } p_3 \text{ is denoted as } bu = 0.0025 + k_3 \times 0.001;\\
&\text{Considered variability range of parameter } p_3 : k_3 = 1 : 3\\[1em]
&\text{Parameter 4: } p_4\\
&\text{The real value of } p_4 \text{ is denoted as } bd = 0.0025 + k_4 \times 0.001;\\
&\text{Considered variability range of parameter } p_4 : k_4 = 1 : 3\\[1em]
&\text{Parameter 5: } p_5\\
&\text{The real value of } p_5 \text{ is denoted as } SL = k_5 \times 0.0005;\\
&\text{Considered variability range of parameter } p_5 : k_5 = 1 : 2\\[1em]
&\text{Parameter 6: } p_6\\
&\text{The real value of } p_6 \text{ is denoted as } k = k_6;\\
&\text{Considered variability range of parameter } p_6 : k_6 = 2 : 6
\end{aligned} \tag{16}$$

For the above parameters variability ranges, in order to generate the best cumulative profit (see Fig. 7), it was necessary to carry out $K_1 \times K_2 \times \cdots \times K_6$ simulations, in this case $3 \times 3 \times 3 \times 3 \times 2 \times 5 = 810$.

$$Zsk(p1_{opt}, p2_{opt}, \ldots, p6_{opt}) = max(Zsk(p1_{k_1}, p2_{k_2}, \ldots, p6_{k_6}) \tag{17}$$

for $k_1 = 1, 2, \ldots, K_1; k_2 = 1, 2, \ldots, K_2; \ldots; k_6 = 1, 2, \ldots, K_6$.

In the MA2 algorithm Zrec variable corresponds to the Zsk for the optimal parameter values. The algorithm completion time was approximately 160 s (on an average personal computer in 2014). The best final result reached Zrec = 0.3300 (3,300 pips—after 5,000 h).

The difference between Figs. 6 and 7 is shocking. This demonstrates the usefulness of searching for unexpected solutions often radically changing the result of trading.

4 MA3—Grounds for the Volatility of the Optimal Parameters in Time

One more test was conducted using MA3 algorithm. The range of variability of individual parameters was greatly widened, so that the execution time of all the research was 3,800 s (in the previously described conditions). There has been a much

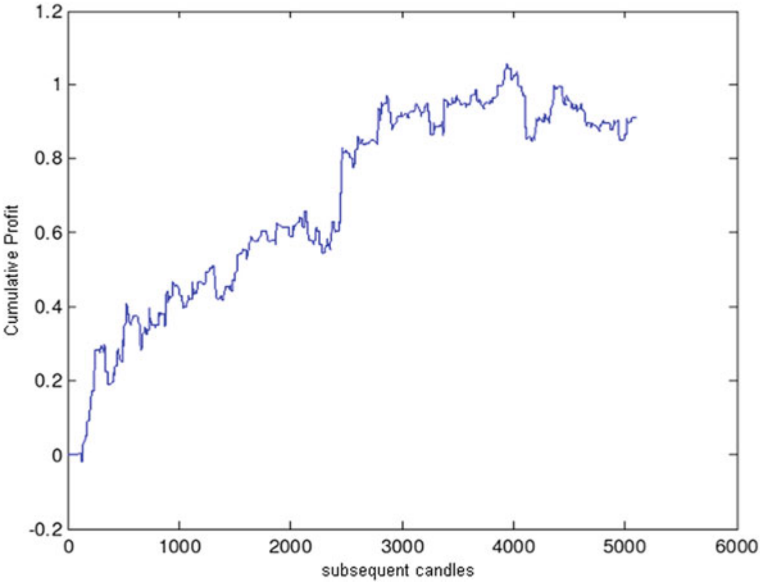


Fig. 8 The cumulative profit for the MA2 algorithm, 5,000 candles and after extension of ranges of parameters variability

better final result Zrec = 0.9107 and a smoother trajectory of capital growth (see Fig. 8).

Obviously, it should be remembered that this extremely beneficial trajectory of cumulative profit relates to a constant period of time, for which those best parameter values were “learned”. This does not mean that in the future, for the candles with indices greater than 5,000, the effectiveness of the strategy will continue, it is only probable.

Thus appears the question of how to change the parameters of the strategy, how often, how long, and on which section of the time series to search for them, in order to adapt the strategy to the changing data environment.

Therefore, appropriate changes have been made in the MA3 algorithm, in order to adapt it to the machine learning. The MA4 algorithm takes into account these changes. First, it was checked, whether there is actually a large variability of the optimal parameters in the subsequent time series periods. For this purpose, the algorithm was investigated with the same data (EURUSD 1 h). It was investigated how the optimal parameters vary for subsequent periods of time series: each period began every 100 candles and was investigated for 20 candles. It is a relatively small period of time (20 h is less than day) and the values of optimal parameters should be changing in a relatively smooth way, it would appear. Whereas after investigating 10 consecutive periods for I = 1, 2, . . . , 10 and for all 6 parameters j = 1, 2, . . . , 6, the following matrix of parameters indices was obtained, characterized by quite a big variability:

$$P_{ij} = \begin{array}{l} [1\ 2\ 1\ 1\ 3\ 3 \\ \ 1\ 6\ 1\ 1\ 7\ 3 \\ \ 1\ 2\ 1\ 1\ 7\ 3 \\ \ 1\ 7\ 4\ 1\ 3\ 4 \\ \ 4\ 6\ 1\ 1\ 8\ 6 \\ \ 2\ 7\ 1\ 3\ 5\ 6 \\ \ 1\ 7\ 1\ 1\ 5\ 6 \\ \ 4\ 5\ 1\ 1\ 3\ 4 \\ \ 4\ 7\ 1\ 1\ 5\ 4 \\ \ 2\ 2\ 1\ 2\ 3\ 3]; \end{array} \tag{18}$$

Every row of the above matrix represents the coded optimal value of the jth parameter in the ith simulation for investigated period in the nearest future (in this case on 20 last candles). For example, $P_25 = 7$ represents the best value of SL parameter (coded by k_5 in (16)) for investigated $i = 2$ period, it should be interpreted as $SLopt = 7 \times 0.0005 = 0.0035$ (35 pips).

In the case of large variability of parameters (18), the variability was investigated on longer periods of time, equal 100 candles (100 h). Each stage of investigation, as previously, was 100 candles in length. The results are presented as a matrix (19).

$$P_{ij} = \begin{array}{l} [2\ 5\ 1\ 2\ 7\ 3 \\ \ 4\ 6\ 2\ 1\ 8\ 3 \\ \ 4\ 2\ 1\ 3\ 8\ 6 \\ \ 1\ 7\ 5\ 1\ 8\ 5 \\ \ 2\ 2\ 1\ 3\ 8\ 6 \\ \ 4\ 6\ 1\ 3\ 6\ 6 \\ \ 2\ 3\ 5\ 1\ 6\ 6 \\ \ 4\ 7\ 4\ 2\ 5\ 5 \\ \ 4\ 8\ 1\ 1\ 5\ 4 \\ \ 1\ 3\ 1\ 3\ 7\ 3]; \end{array} \tag{19}$$

The variability (within individual columns) is even bigger than in (18).

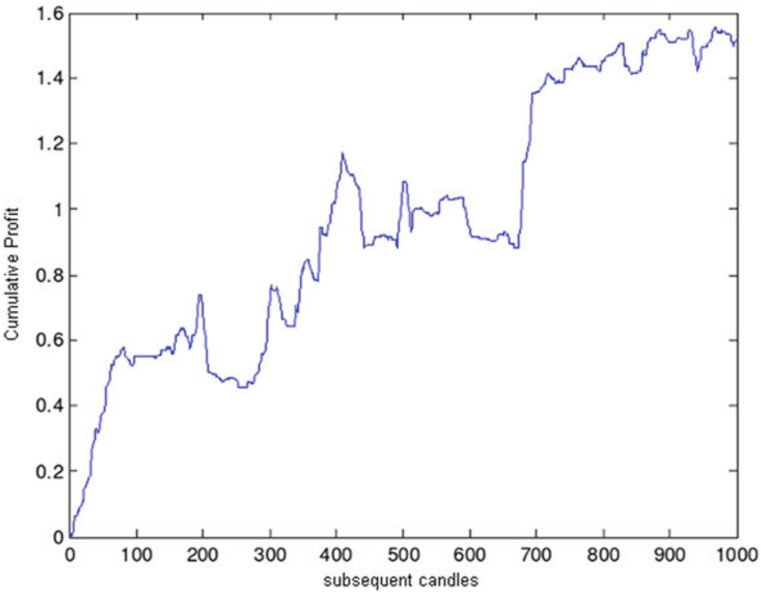


Fig. 9 The curve of cumulative profit for MA4 strategy in machine learning mode for the R = 8/2 ratio

This suggests to cautiously approach the planning of machine learning with the use of longer periods of testing data sets. The increasing variability of the values of optimal parameters with increasing testing period may cause larger errors and lower the efficiency of the strategy.

In summary, the selection of validation windows in machine learning remains an open issue with the suggestion to check the rather short lengths of learning and testing.

In MA4 algorithm, all optimal parameter values are used to calculate the profit in the testing period of time series, following right after the learning period.

According to the above conclusions, the first test was carried out on the period of 1,000 candles, performing 500 tests on two candles each. The parameters were learned on the period of 8 candles before each test. The ratio of the length of the training window to the testing window was denoted as R = 8/2. The result is shown in Fig. 9.

On the period of 1,000 candles a very high final result was obtained, of the order of magnitude of 15,000 pips, with a relatively high drawdown of 3,000 pips, which gives the Calmar ratio of approximately 5 (ratio of profit to the maximum drawdown). Compared with the previous results of learning the optimal parameters for all the learning data (e.g. Figs. 7 and 8), Fig. 9 presents an excellent result.

At this point the question appears, whether there is a possibility to obtain better final result and higher smoothness of profit growth curve for a different ratio between the lengths of learning and testing windows.

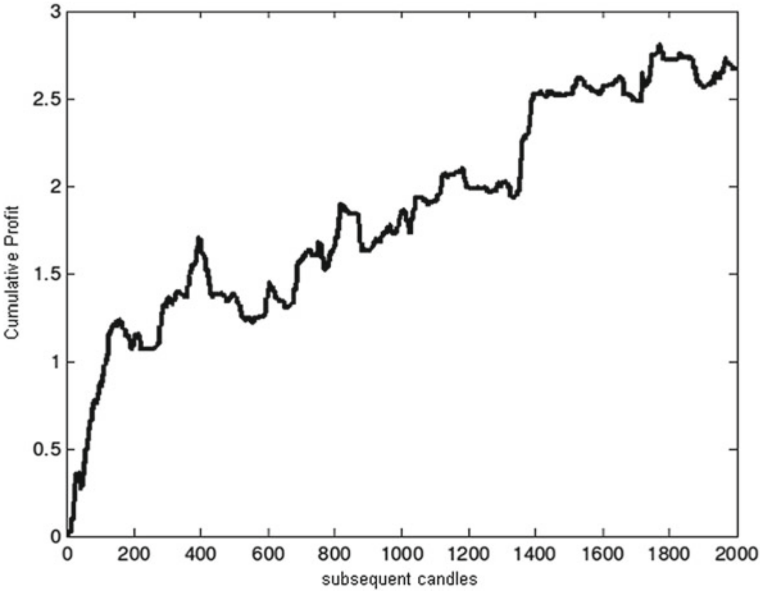


Fig. 10 The cumulative profit for the ratio R = 5/1

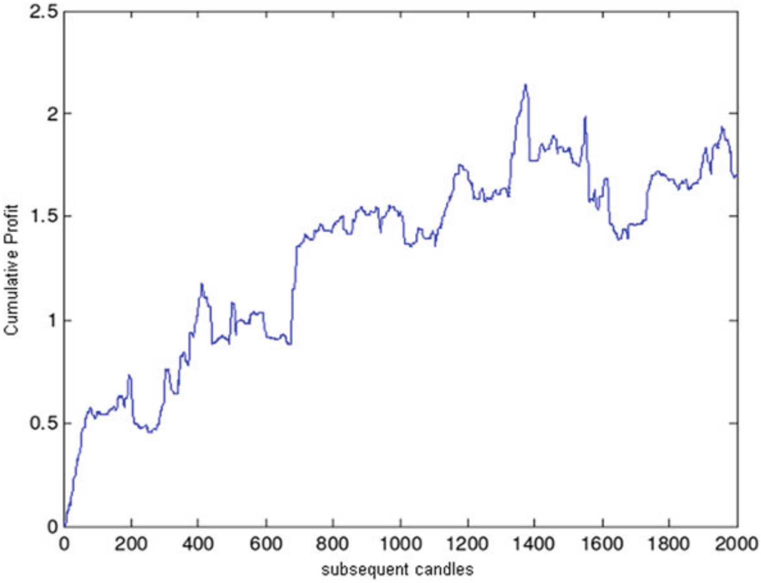


Fig. 11 The cumulative profit for the ratio R = 5/1 and 2,000 candles in the range of candles from 5,000 to 7,000 with extended parameters variability range

The study was carried out for different values of R and for different time horizons.

The results presented in the above figures for the learning/testing ratio of 5/1 and 8/2 are very interesting. No good results were obtained by increasing the learning period, and then the testing period, respectively (see Fig. 14). Hence, it is advisable to often calculate the optimal parameters due to their high variability. Watching the final values on individual graphs of cumulative profits it can be seen, that the average effectiveness of the strategy (understood as an average profit for one candle) is high, at a level of a few to several pips (for one hour). It is also a very good result. Recalling that the pip is a change in the exchange rate (also in the profit measured in the same values) observed on the fourth decimal place (for the EURUSD currency pair considered here), then the final profit equal to e.g. 1.0 is 10,000 pips, the result of 2.0 is 20,000 pips, etc. For example, Fig. 8 presents the final result on cumulative profit curve of approximately 1.0, obtained in simulation on 5,000 candles, thus profit on a candle is approximately equal 2 pips.

In Fig. 9 the profit on a candle is approximately equal to 15 pips, in Fig. 10 about 13 pips, and in Fig. 11 about 9 pips.

5 Conclusions

Are the results presented above exciting? According to the authors, yes. This is the best result among several recently published strategies of the first author [10, 14]. The capital accumulation trajectories presented there may be comparable in terms of

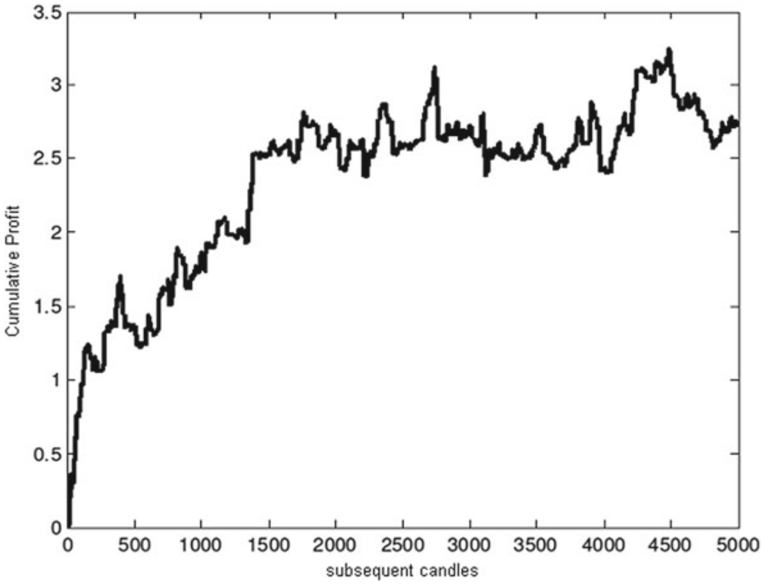


Fig. 12 The cumulative profit for the ratio R = 8/2 and 2,000 candles for EURUSD 1 h

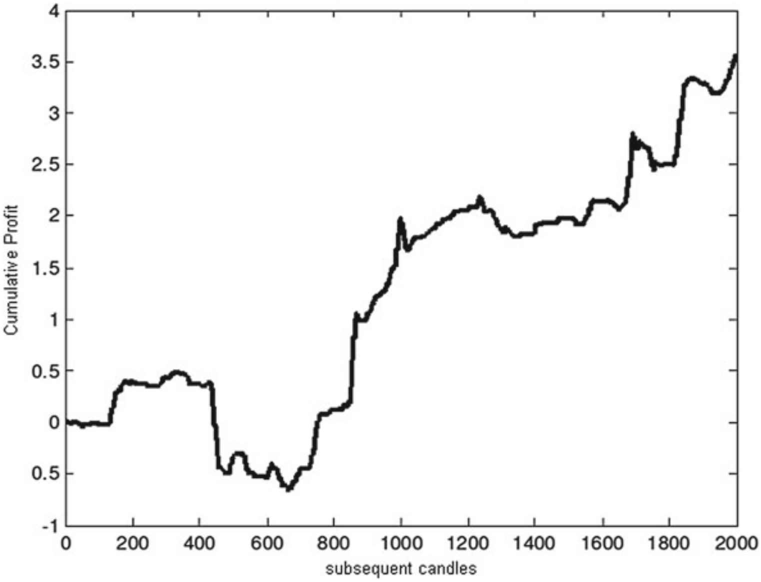


Fig. 13 The cumulative profit for the ratio R = 8/2 and 2,000 candles

the risk management (Calmar ratios are similar), but the profit per candle is definitely lower for comparable data sets.

The results are also interesting due to the relatively low capital drawdown value, approximately 20 % of the final profit.

In the trading practice on the brokerage platforms it can indicate big profits even with a very careful investor's behavior. For example, let us assume the initial capital of 100,000 USD and the leverage of 100:1. If the investor trades cautiously, putting 100 USD into each opened position (i.e., using only 0.1 % of the capital), after 5,000 one hour candles (almost a year) approximately 27,000 pips of profit may be gained (see Fig. 12). With the given 100:1 leverage (1 pip equals 1 USD then) this amounts to 27,000 USD of profit, with a maximum drawdown of 20 %, i.e., 5,500 USD. In this example, the profit after a year would reach approximately 30 % of the initial capital. If the investor is more inclined to take a risk (smaller initial capital), the relative profit may be larger (Fig. 13).

This is a relatively simple investment strategy, with only a few parameters, easy to implement within the brokerage platform, e.g., after the conversion to MQL (MetaTrader) or EasyLanguage (TradeStation). The strategy was tested on data not used in the learning process, and thus it meets the basic condition of the scientific soundness. The simulations using EURUSD 1 h currency pair indicate, that the optimal lengths of learning and testing windows span only several candles, which is an important finding for this strategy. Any attempts to obtain a good results for the larger lengths of the periods gave far worse results (see Fig. 14).

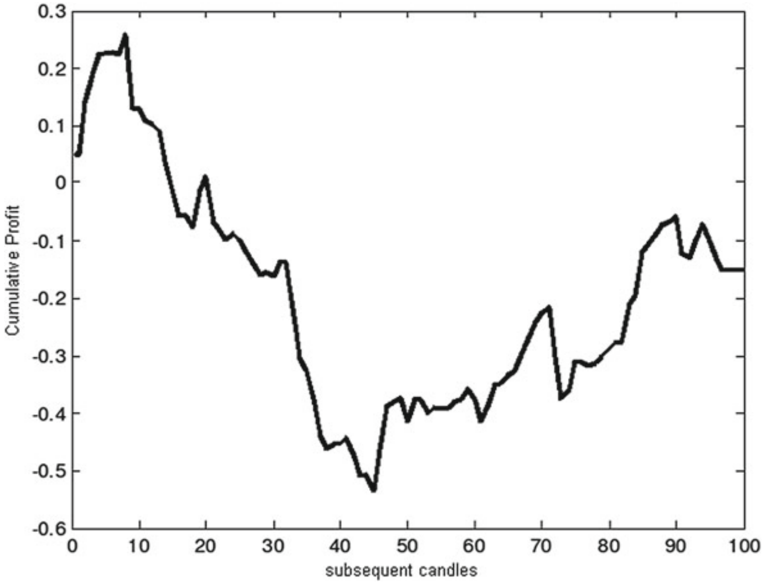


Fig. 14 The cumulative profit for the ratio R = 80/15. The result is completely unsatisfactory

In all the simulations carried out over a short learning and testing periods, the cumulative profit trajectories showed that the so-called Calmar criterion reached values of the order of magnitude of 5 (the ratio of the final profit to the largest drawdown of capital within a given curve). According to the authors, it is also a very good result, which could be acceptable also for investors with high risk aversion. It is also important to note the impact of machine learning on the results of the simulation. Using machine learning better results were obtained, than by finding the optimal parameters for the whole period of time—compare Figs. 8 and 12. The results of the test of strategy's predictive abilities, expressed by the confrontation between the result in Figs. 5 and 6, are also interesting. This comparison indicates the ability to predict [16] even with the poor initial parameter space. Further research could be directed at extending the simulations onto other markets (starting with currency pairs) and other sampling rates (candles periods).

Conflict of interest The authors declare that there is no conflict of interest regarding the publication of this paper.

References

1. Brock, W., Lakonishok, J., LeBaron, B.: Simple technical trading rules and the stochastic properties of stock returns. J. Financ. **47**(5), 1731–1764 (1992)
2. Tian, G., Wan, G., Guo, M.: Market efficiency and the returns to simple technical trading rules: new evidence from U.S. equity market and Chinese equity markets. Asia-Pac. Financ. Mark. **9**(3–4), 241–258 (2002)

3. Muriel, A.: Short-term predictions in forex trading. Phys. A Stat. Mech. Appl. **344**(1), 190–193 (2004)
4. Gencay, R.: Linear, non-linear and essential foreign exchange rate prediction with simple technical trading rules. J. Int. Econ. **47**(1), 91–107 (1999)
5. Cai, B., Cai, C., Keasey, K.: Market efficiency and returns to simple technical trading rules: further evidence from U.S., U.K., Asian and Chinese stock markets. Asia-Pac. Financ. Mark. **12**(1), 45–60 (2005)
6. Krutsinger, J.: Trading Systems: Secrets of the Masters. McGraw-Hill Inc., New York (1997)
7. Elder, A.: Come Into My Trading Room. Wiley Trading, New York (2002)
8. Owens, S., Lizotte, O.: When to Trade. FX Engines (2004)
9. Friesen, G.C., Weller, P.A., Dunham, L.M.: Price trends and patterns in technical analysis: a theoretical and empirical examination. J. Bank. Financ. **33**(6), 1089–1100 (2009)
10. Wiliński, A., Nyczaj, T., Bera, A., Błaszyński, P.: A study on the effectiveness of investment strategy based on the concept of pivot points levels using Matthews criterion. J. Theor. Appl. Comput. Sci. **7**(4), 42–55 (2013)
11. Wiliński, A., Bera, A., Nowicki, W., Błaszyński, P.: Study on the effectiveness of the investment strategy based on a classifier with rules adapted by machine learning. ISRN Artif. Intell. **2014**, 1–10 (2014)
12. Satchwell, C.: Pattern Recognition and Trading Decisions. McGraw-Hill, New York (2005)
13. Raghuraj, R., Lakshminarayanan, S.: Variable predictive models-A new multivariate classification approach for pattern recognition applications. Pattern Recognit. **42**(1), 7–16 (2009)
14. Wiliński, A., Bera, A., Błaszyński, P., Jarlaczynski, M.: The investment strategy based on behaviour of artificial earthworm for use in algotrading. In: Proceedings ITISE International Work-Conference on Time Series, vol. 2, pp. 991–1005. Granada (2014)
15. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Berlin (2006)
16. Marjak, H.: The architecture selection for neural network—a financial criterion or a prediction criterion. Pol. J. Environ. Stud. **17**(3b), 255–261 (2008)

Pattern Recognition in the Japanese Candlesticks

Leszek Chmielewski, Maciej Janowicz, Joanna Kaleta and Arkadiusz Orłowski

Abstract Pattern recognition analysis based on k-nearest neighbors classifiers is applied to the representation of the stock market dynamics with the help of the Japanese candlesticks augmented by the accompanying volume of transactions. Examples from a post-emerging Warsaw stock market are given. Conditions under which the Japanese candlesticks appear to have a reasonable predictive power are provided. The dependence of the results on the number of nearest neighbors, the length of the candlestick sequence, and the forecast horizon are shown. Possible ways of the forecast improvement are discussed.

Keywords Pattern recognition · Stock market forecast · Japanese candlesticks · k-Nearest Neighbors

1 Introduction

The problem of forecasting in the time series containing a stochastic component belongs to the most fascinating as well as practically important issues in the study of dynamics of natural, social, and economical systems. Among the various methods of analysis of stochastic time series one should mention very traditional ones, based on

L. Chmielewski · M. Janowicz · J. Kaleta · A. Orłowski (✉)
Faculty of Applied Informatics and Mathematics (WZIM),
Warsaw University of Life Sciences (SGGW),
Ul. Nowoursynowska 159, 02-775 Warsaw, Poland
e-mail: leszek_chmielewski@sggw.pl
URL: http://www.wzim.sggw.pl

M. Janowicz
e-mail: maciej_janowicz@sggw.pl

J. Kaleta
e-mail: joanna_kaleta@sggw.pl

A. Orłowski
e-mail: arkadiusz_orlowski@sggw.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_19

the broadly understood concept of regression, as in, e.g., [1], those based on linear filters, see, e.g., [2], as well as the methods associated with the powerful concept of state-space [3].

Methods and techniques of pattern recognition and data mining are relatively new in the field, but the last decade of the previous century brought a real outburst of research in this area. We mention a few papers associated with indexing [4–6], clustering [7, 8], classification [9, 10], and anomaly detection [11, 12], see [13] for a more detailed bibliography. There exist three more or less standard categories into which the methods of pattern recognition and data mining usually fall: supervised learning, which includes classification, unsupervised learning (pattern detection, clustering, class discovery, characterization, change detection, and Fourier, wavelet, and principal component decomposition among other things), and semi-supervised learning. The survey paper [13] contains a useful exposition of the above components.

In this contribution, we study the classification of patterns in the Japanese candlesticks representation of the time series, which appear in stock markets. Thus, our study is devoted to the *supervised learning* in the broad sense. More precisely, we perform the candlestick patterns classification using the k-nearest neighbor classifier. The emerging patterns are then analyzed from the point of view of their predictive power (or lack of it). From this point of view, our work may be viewed as a contribution, which accompanies that of [14], in which the machine learning approach has been applied to develop an efficient investment strategy.

The main body of this work is organized as follows. In Sect. 2 we recall the definition of the Japanese candlesticks as well as what we call *augmented candlesticks*, which include information about the volume of transactions. Distances between the sequences of candlesticks are also defined. In Sect. 3 we present our approach to the pattern recognition problem in the time series associated with stock markets. Section 4 contains a discussion of the predictive power of the sticks. Finally, Sect. 5 comprises some concluding remarks.

2 Japanese Candlesticks as a Representation of Value of Assets in Stock Market

The Japanese candlestick is a 4-tuple $(O(a,t), X(a,t), N(a,t), C(a,t))$, where O denotes the opening value of the asset a at the trading day t, X is the maximum value (*high*) reached during the trading session, N is the minimum (*low*), and C is the closing value.

In what follows below, we employ five elements (O, X, N, C, V) which we call an *augmented Japanese candlestick*, where V represents the transaction volume associated with the asset and the trading day. An augmented candlestick of the asset a on the day t can be denoted as a 5-tuple

$$L(a,t) = (O(a,t), X(a,t), N(a,t), C(a,t), V(a,t)). \tag{1}$$

In the following, we shall call it simply a candlestick. The time series of $n + 1$ candlesticks, called otherwise a sequence, can be written as

$$S_n(a, t) = (L(a, t), L(a, t + 1), \dots, L(a, t + n)). \tag{2}$$

Each sequence has its starting time t and ending time $t + n$. The sequences having length from 1 to 5 are investigated below.

We define the distance between two candlesticks as

$$D(a, t_1, t_2) = \sqrt{\sum_{A \in Z} (A(a, t_1) - A(a, t_2))^2}, \tag{3}$$

where $Z = \{O, X, N, C, V\}$. In order to consider this formula meaningful, the values of the asset and the transaction volume must be comparable. To achieve this, we normalize all time series by subtracting the closing values from the opening ones as well as from the maxima and minima, and dividing O, X, N, and C by the standard deviation of C. Similarly, the volume is also divided by its standard deviation. In this way, the standard deviations of renormalized C and V are exactly 1. All time series analyzed further are normalized in the above sense.

3 Pattern Recognition with the Help of *k*-Nearest Neighbor Classifier

The above concepts have been applied by us to the share prices of the Warsaw stock market.

A sequence as defined by Eq. (2) will be treated as a pattern. If the typical pattern recognition nomenclature is used, the feature set of this pattern is formed by $n + 1$ sets, each being a 5-tuple of the elements of a candle.

For each two sequences, a distance between them can be defined in the natural (Euclidean) way:

$$D_n(a, t_1, t_2) = \sqrt{\sum_{i=0}^{n} D^2(a, t_1 + i, t_2 + i)}. \tag{4}$$

For each sequence $S_n(a, t)$ we can define a preceding sequence $S_n(a, t')$ where $t' < t$. The sequences preceding a given sequence can have common days with it. With the help of the distance D_n, for each sequence $S_n(a, t)$ we can identify k preceding sequences which are nearest to it in the sense of minimizing the distance D_n. Such k preceding, nearest sequences form the set of *nearest neighbors* of S.

We have considered the set of 20 assets belonging to the so-called WIG20, the group of the largest companies of the Warsaw stock market—Polish “blue chips.” For each sequence in this set, we have found its k nearest neighbors among the preceding sequences for the same asset.

To facilitate the search for nearest neighbors, the elements O, X, N, and C of each candlestick have been translated by the constant B chosen in such a way that

$$O(a,t') - B = O(a,t). \tag{5}$$

Thus, the compared sequences have equal first elements.

For each sequence S, for given n and k, we have calculated k starting times t'_l, $l = 1, \ldots, k$, and the corresponding distances from the considered sequence.

4 Forecasts Using Patterns in Candlesticks: Examples

Quite intuitively, the above notions and procedure can serve the purpose of an elementary forecasting. Indeed, let $S_n^{(l)}(a, t'_l), l = 1, 2, \ldots, k$, be one of the k nearest neighbors of $S_n(a,t)$ of length $n+1$. We are interested in a prediction of the closing value $C(a, t+m)$, where $m > n$. Proceeding along a rather well-established route, we can associate with every distance $D_n^{(l)}(a, t, t'_l)$ between S and $S^{(l)}$ a weight

$$W_n^{(l)}(a,t,t'_l) = 1/D_n^{(l)}(a,t,t'_l) \tag{6}$$

and form a prediction $\bar{C}_n(a, t+m)$ as the weighted sum:

$$\bar{C}_n(a,t+m) = \sum_{l=1}^{k} W_n^{(l)}(a,t,t'_l)C(a,t'_l+n). \tag{7}$$

The prediction will be called *correct* if

$$\operatorname{sign}(\bar{C}_n(a,t+m) - C(a,t+n)) = \operatorname{sign}(C(a,t+m) - C(a,t+n)), \tag{8}$$

otherwise it will be called *incorrect*. We have also considered a modification of the above definitions, namely an idea to use the candles to predict the *mean* change during the following N trading sessions. More precisely, let C_N denote the following average of the closing values:

$$C_N(a,t) = \frac{1}{N}\sum_{p=0}^{N-1} C(a,t+p). \tag{9}$$

We define the prediction of the above average as $\sum_{l=1}^{k} W_n^{(l)}(a,t,t'_l)C_N(a,t'_l)$, with the understanding that all t'_l are not greater than $t - N$. Again, the prediction will be correct if the sign of the predicted mean value is the same as the sign of the actual mean value.

To measure the quality of predictions given by augmented candlesticks, one can introduce the following simple function:

$$Q = \frac{P_{\text{correct}} - P_{\text{incorrect}}}{P_{\text{correct}} + P_{\text{incorrect}}}, \tag{10}$$

where P_{correct} ($P_{\text{incorrect}}$) is the total number of correct (incorrect) predictions.

Tables 1, 2, 3, and 4 contain the results for two shares recorded in the Warsaw stock market: Alior Bank and KGHM Polish Copper. The results for the function Q are shown as dependent on k (the number of nearest neighbors) and N for two different lengths of the candlesticks sequences, 1 and 5 (we have also obtained results for intermediate lengths but do not present them as they are not particularly illuminating). Predictions have been made for the mean of the N values of the closing value C following the ending time of a sequence of candlesticks.

Table 1 Prediction quality function Q as dependent on the number of nearest neighbors k and the forecast depth N for the shares of Alior Bank

$N\backslash k$	1	2	3	4	5	6	7	8
1	−0.099	−0.072	−0.046	−0.059	−0.013	−0.039	−0.066	−0.052
2	0.063	0.063	0.003	0.003	−0.056	−0.036	0.003	0.029
3	0.013	0.046	0.093	0.060	0.033	0.033	0.033	0.06
4	0.023	0.036	0.130	0.117	0.043	0.083	0.050	0.030
5	0.040	0.040	0.087	0.100	0.020	0.060	0.020	0.046
6	0.070	0.070	0.097	0.104	0.037	0.030	−0.023	0.016
7	0.074	0.074	0.135	0.108	0.027	0.033	0.000	−0.020
8	0.084	0.084	0.071	0.057	−0.023	0.010	−0.010	−0.037

The sequences of candlesticks consist of one element

Table 2 Prediction quality function Q as dependent on the number of nearest neighbors k and the forecast depth N for the shares of Alior Bank

$N\backslash k$	1	2	3	4	5	6	7	8
1	−0.067	−0.040	−0.053	0.013	−0.100	−0.026	−0.134	−0.026
2	−0.050	−0.043	0.043	0.016	−0.023	−0.010	0.003	−0.023
3	−0.094	−0.081	−0.060	−0.033	0.013	−0.013	−0.006	−0.060
4	−0.057	−0.044	0.003	0.010	−0.030	0.010	0.016	0.003
5	−0.013	−0.013	−0.020	−0.020	−0.040	0.000	−0.047	−0.006
6	−0.030	−0.030	0.030	0.030	−0.023	−0.023	0.017	0.010
7	−0.020	−0.020	0.013	0.020	−0.047	−0.027	0.020	−0.020
8	−0.044	−0.044	−0.003	0.003	−0.072	−0.051	−0.058	−0.099

The sequences of candlesticks consist of five elements

Table 3 Prediction quality function Q as dependent on the number of nearest neighbors k and the forecast depth N for the shares of KGHM Polish Copper

$N\backslash k$	1	2	3	4	5	6	7	8
1	−0.101	−0.060	−0.049	−0.057	−0.059	−0.048	−0.057	−0.051
2	−0.024	−0.011	−0.015	−0.001	−0.010	−0.021	−0.020	−0.005
3	−0.003	0.002	−0.001	0.007	−0.001	−0.011	0.003	0.011
4	0.014	0.021	0.016	0.018	0.006	0.009	0.008	0.019
5	0.017	0.021	0.019	0.022	0.002	0.010	0.007	0.005
6	0.012	0.014	0.016	0.020	0.014	0.009	0.016	0.010
7	0.012	0.016	0.006	0.006	0.016	0.011	0.018	0.011
8	0.003	0.004	−0.002	0.000	0.010	0.012	0.014	0.011

The sequences of candlesticks consist of one element

Table 4 Prediction quality function Q as dependent on the number of nearest neighbors k and the forecast depth N for the shares of KGHM Polish Copper

$N\backslash k$	1	2	3	4	5	6	7	8
1	−0.107	−0.058	−0.041	−0.038	−0.039	−0.039	−0.034	−0.031
2	−0.032	−0.019	−0.004	−0.011	−0.007	−0.009	0.002	0.000
3	−0.006	0.005	0.015	0.014	0.026	0.016	0.023	0.016
4	−0.010	−0.005	0.016	0.011	0.007	−0.006	−0.001	0.006
5	0.001	0.009	0.025	0.021	0.029	0.008	0.008	0.011
6	−0.015	−0.013	0.006	−0.003	−0.011	−0.015	−0.006	−0.011
7	−0.007	−0.001	0.012	0.001	0.003	−0.001	−0.010	0.000
8	−0.004	−0.002	0.016	0.007	0.014	0.014	0.004	0.016

The sequences of candlestick consist of five elements

The obvious conclusion one can draw from Tables 1, 2, 3, and 4 is that the overall predictive power of the Japanese candlesticks is very well approximated by zero. In fact, of many thousands of numbers like the above that we obtained for Q from our numerical procedures, there have been only a few larger than 0.1. What is perhaps a little astonishing is the fact that forecasts based on a single candlestick may actually work better than those based on five sticks.

The situation changes somewhat, however, if we allow for a more modest forecasting principle. The procedure applied above can be summarized as an attempt to predict the change in the value of the asset with respect to the close value of the last known candlestick. However, what if we want to predict the change with respect to the *first* (i.e., the earliest) candlestick in a sequence? We have observed the improvement in the forecasting quality even in the case of three sticks, but it is of course better visible in the case of five, as specified in Tables 4 and 5.

The reason for the improvement of the results in Table 5 with respect to that of Tables 1, 2, 3, and 4 is that many of the sequences are parts of approximately linear trends. It is, naturally, far easier to achieve better quality of predictions if one refers

Table 5 Prediction quality function Q as dependent on the number of nearest neighbors k and the forecast depth N for the shares of KGHM Polish Copper

$N\backslash k$	1	2	3	4	5	6	7	8
1	0.171	0.194	0.185	0.197	0.184	0.190	0.208	0.218
2	0.179	0.185	0.198	0.207	0.199	0.207	0.204	0.217
3	0.161	0.162	0.180	0.190	0.191	0.194	0.192	0.201
4	0.160	0.163	0.162	0.184	0.183	0.190	0.191	0.197
5	0.143	0.143	0.153	0.167	0.161	0.167	0.184	0.179
6	0.145	0.146	0.147	0.165	0.150	0.159	0.167	0.165
7	0.151	0.151	0.136	0.160	0.151	0.159	0.153	0.162
8	0.151	0.151	0.146	0.165	0.160	0.171	0.162	0.174

The sequences of candlestick consist of five elements. Predictions have been calculated with respect to the first (earliest) of the candlesticks

to the starting or early points in a trend, instead of a point in the middle of an ascent or a descent.

5 Concluding Remarks

In this work, we have applied the k-nearest neighbors classifier to the patterns that emerge in the Japanese candlesticks being a representation of the state of share prices in the stock market. The candlesticks have been augmented to include information about the volume of transaction in the market for a given asset. The discovered patterns have been used to check the possible predictive power of the candlesticks.

Our results support, in general, the highly skeptical evaluation of the possibility to exploit correlations in the share prices for making profits, although Table 5 strongly suggests that one still can succeed if proper questions are asked as regards the forecast.

We must admit here that the way the technical analysts use the Japanese candlesticks in their studies of share prices dynamics is quite different from that adopted here. Indeed, they use only some very special combinations of the sticks, the most significant ones, for their trading tactics. Also, the similarity between the patterns strongly depends on the context (i.e., the general situation on the market). We plan to investigate these matters in the further work.

References

1. Anderson, T.W.: The Statistical Analysis of Time Series. Wiley, New York (1971)
2. Haykin, S.: Adaptive Filter Theory. Prentice-Hall, Englewood Cliffs (1986)
3. Durbin, J., Koopman, S.J.: Time Series Analysis by State Space Methods. Oxford University Press, Oxford (2001)

4. Agrawal, R., Psaila, G., Wimmers, E.L., Zait, M.: Querying shapes of histories. In: Proceedings of the 21st International Conference on Very Large Databases. Zurich, 11–15 September 1995
5. Camerra, A., Palpanas, T., Shieh, J., Keogh, E.: iSAX 2.0: Indexing and mining one billion time series. In: 2010 IEEE International Conference on Data Mining, ICDM, pp. 58–67 (2010)
6. Keogh, E., Chakrabarti, K., Pazzani, M.: Locally adaptive dimensionality reduction for indexing large time series databases. In: Proceedings of ACM SIGMOD Conference on Management of Data, Santa Barbara, 21–24 May 2001
7. Keogh, E., Pazzani, M.: An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining. New York, 27–31 August 1998
8. Liao, T.W.: Clustering of time series data—a survey. Pattern Recognit. **38**, 1857 (2005)
9. Geurts, P.: Pattern extraction for time series classification. In: Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery, Freiburg (2001)
10. Radovanovic, M., Nanopoulos, A., Ivanovic, M.: Time-series classification in many intrinsic dimensions. In: Proceedings of the SIAM International Conference on Data Mining, SDM, Columbus, Ohio, 29 April–1 May 2010
11. Keogh, E., Lin, J., Fu, A.W.: HOT SAX: Efficiently finding the most unusual time series subsequence. In: Proceedings of the 5th IEEE International Conference on Data Mining, Houston, 27–30 November 2005
12. Preston, D., Protopapas, P., Brodley, C.: Event discovery in time series. In: Proceedings of the SIAM International Conference on Data Mining, SDM, Sparks, Nevada, 30 April–2 May 2009
13. Lin, J., Williamson, S., Borne, K., DeBarr, D.: Pattern recognition in time series. In: Way, M., Scargle, J.D., Kamal, M.A., Srivastava, A.N. (eds.) Advances in Machine Learning and Data Mining in Astronomy. J. Chapman and Hall/CRC Press, Boca Raton (2012)
14. Wiliński, A., Zabłocki, M.: The investment strategy based on the difference of moving averages with parameters adapted by machine learning. In: Wiliński, A., El Fray, I., Pejaś, J. (eds.) Soft Computing in Computer and Information Science, Advances in Intelligent Systems and Computing Series, Vol. 342, Springer, pp. 207–225 (2015)

Part III
Information Technology Security

ProToc—An Universal Language for Security Protocols Specifications

A. Grosser, M. Kurkowski, J. Piątkowski and S. Szymoniak

Abstract This paper shows a new language for security protocols specifications. First, we present other specification languages. As far as the use is concerned, Common Language and its restrictions are presented. Then, CAPSL language is shown and introduced within the AVISPA Project, HLPSL Language. The paper ends with the original approach toward protocol specifications, which is a new ProToc language as well as its grammar and examples of protocols specifications in the language. ProToc has been used as the language of specification for the tool of automatic verification of concurrent systems VerICS.

Keywords Security protocols · Formal verification · Specification languages

1 Introduction

One of the tools that take advantage of cryptographic techniques to achieve goals connected with widely understood security in computer networks is the so-called security protocols. These protocols are a subclass of a class of general-purpose

The author acknowledges the support of the FNR (National Research Fund) Luxembourg under project GALOT—INTER/DFG/12/06.

A. Grosser, J. Piątkowski · S. Szymoniak (✉)
Instytut Informatyki Teoretycznej I Stosowanej, Politechnika Częstochowska Dąbrowskiego 69/73, 42-200 Częstochowa, Poland
e-mail: sabina.szymoniak@icis.pcz.pl

A. Grosser
e-mail: andrzej.grosser@icis.pcz.pl

J. Piątkowski
e-mail: jacek.piatkowski@icis.pcz.pl

M. Kurkowski
Computer Science and Communication Group, University of Luxembourg, 6, Rue Richard Coudenhove-Kalergi, 1359 Luxembourg, Luxembourg
e-mail: miroslaw.kurkowski@uni.lu

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_20

communication protocols. The history of constructing and using the protocols is interesting and educative. Everything began together with creating and development of computer networks. In 1978 Needham and Schroeder in a paper [1], trying to solve problems related to authentication entities in networks as well as ensuring confidentiality of forwarded data, offered a few concurrent algorithms (protocols) whose goal was to achieve presumed security objectives. These algorithms were used in practice quickly.

Within the next years several security protocols were created which fulfilled various security objectives in computer networks. Today, the protocols are commonly used during communication in ICT technologies and computer systems. Usually, they are component parts of used communication protocols. The examples of these may be as follows: Kerberos system, SSL/TLS, SESAME, and WPA. Further information may be found in [2, 3].

Unfortunately, within years it turned out that the algorithms could be deceived and generally they do not provide a desired level of protection. Simple but supposedly proper protocol constructions proved to be easily attacked by, gently speaking, a mean user, called hereinafter an Intruder [4]. Thus, it was necessary to establish methods of protocol correctness verification.

Certainly, the problem of protocol correctness was immediately used as crucial and intensive research on this problem commenced. Security protocols are distinctive due to the fact that they are made by network users (agents, computers, servers) repeatedly, sometimes even several thousand times per day and usually concurrently. Implementation of protocols depends on information from knowledge database, e.g., repositories of encryption key certificates.

The use of security protocols to guarantee set security objectives in networks and computer systems requires particular caution as far as correctness of functioning is concerned. Incorrect protocols functioning may lead to various users' losses. It should be remembered that wrong construction of an algorithm and wrong implementation of a correct algorithm are two different things. Due to the fact that protocols are usually short and simple, initially informal argumentation was used in order to prove that they work properly and to convince themselves and other people that in fact they do what they are expected to do.

Certainly, formal methods of testing the properties of protocols are used as well. In short, mathematical models of implementation of protocols are constructed. The models are later properly tested. However, in order to verify such a protocol, its formal specification is necessary. Thus, properly formalized specification languages of described protocols are needed.

This article presents the current and new methods used for specification of security protocols. At first, the CL, mentioned in the literature is presented as well as the opportunities and restrictions of this language. Then, CAPSL (Common Authentication Protocol Specification Language) is presented. Also, HLPSL (High-Level Protocol Specification Language) introduced within the AVISPA Project is described. The article ends with a new, original approach to protocols specification, namely ProToc language. The grammar of the language and examples of specifications of protocols in this language have been used as specification language for a tool of automatic verification of VerICS concurrent systems [5–10].

2 Languages of Specification

2.1 Common Language

Common Language is the simplest and the earliest launched language for specification of security protocols. It was used to describe protocols from the first works connected with them [1]. Unfortunately, it was never formalized. The grammar of the basic version is not much complicated: a protocol is described as the series of steps, described by stating a sender of the message, its recipient, and the content of the sent letter. The step is specified in the following way:

`A -> B : M`,

where `A` is a sender, `B` is a recipient, and `M` is the message. The grammar of the message is as follows:

`M : A | K | T | L | M_1, M_2 |` $\texttt{<M>}_{\texttt{K}}$,

where `A` belongs to the considered set of system users, `K` to the set of cryptographic keys, `T` to the set of time stamps, `L` stands for lifetime—the period of validity of time stamp. The cryptographic keys used in specification may be certainly symmetric and asymmetric ones. In the former case, keys are marked as `K_AB`, where A and B are users sharing the key, in the latter, symbol `K_A` stands for a public key A, and $\texttt{K\_A}^{-1}$ stands for its private key. `M_1`, `M_2` mean simply concatenation of two messages `M_1` and `M_2`, and by transcript $\texttt{<M>}_{\texttt{K}}$ means cryptogram which is encrypted by the key `K` containing message `M`.

CL language is very simple and presumably it is hard to imagine a simpler language for protocols specification. However, it is worth noticing that it requires additional information, concerning, for example, a description of the internal actions during implementation of protocol, including generating new elements such as keys, *nonces* (quasi-random numbers generated to the needs of one session) or time stamps. There is also no information about which data a sender should use to create a message.

2.2 CAPSL

CAPSL (Common Authentication Protocol Specification Language) is a high-level specification language dedicated to specification of security protocols, their properties, and correctness. The language was suggested by Millen as an input language for NRL tool [11–13]. The description may be found in [14–16]. It may be stated that CAPSL is an essential extension of CL. In contrast to CL, it is a fully formalized language. Selected, interesting for specification of security protocols aspects of CAPSL are presented below. As an example, later in the article the specification of NSPK protocol (Needham Schroeder Public Key Authentication Protocol) is discussed.

CAPSL is a more valuable language than CL, which facilitates a full specification of a protocol as well as a formal stating of the kind of cryptography used in a protocol and specifying which information is newly generated. The possibility of formal specification of expected and possibly tested protocols properties is essential.

2.3 AVISPA Project and HLPSL Language

Currently, the most widely acclaimed system for formal verification of security protocols in the world is the AVISPA (Automated Validation of Internet Security Protocols and Applications) system [17]. The system resulted from a cooperation of several scientific environments, i.e., Universities in Genoa, Zurich, Nancy, and a Siemens branch in Munich. It was necessary for the protocol to create a special, role-based high-level programming language HLPSL (High-Level Protocol Specification Language) whose aim is to specify tested security protocols [18].

The AVISPA system is quite a complex system. Before entering, a specified in HLPSL protocol must be given.The specification is then translated into intermediate language, of low level, so-called `Intermediate Format (IF)`. The verification is based on the use of one out of four modules offered in a tool. Unfortunately, so far the AVISPA project has not supported testing the protocols including time parameters. The process of improving the expressiveness of a language by adding time was also made in Poland. The intruder model is formalized as a set of activities that it may perform. AVISPA focuses on the well-known model Dolev–Yao [4].

The specification language HLPSL used in the AVISPA system is a high-level specification language, based on roles. For each participant of a protocol, one basic role is defined in which particular parameters connected with a participant's actions while implementation of protocol are described. Particular basic roles describe then what kind of information may be used by a given participant initially `(parameters)`in its initial state of knowledge `(initial state)` and the way in which a given state may undergo changes in the so-called transition `(transitions).`

A given specification may be used later by one or several users playing a given role. Then, to initiate given roles we should describe how various users communicate with one another by multiple "splicing" basic roles into composed roles. Thus, we obtain a specification of a scheme of data exchange while verifying the protocol.

The specification also describes additional parameters of verification. Is it stated in an input file what safety properties should be tested as well as the size of the tested areas of protocols. Declaration and stating the aims which should be fulfilled by protocols and which will be tested during verification takes place in another special part.

To sum up, HLPSL [18] is a complex language enabling a full specification of cryptographic protocols. It has been created especially to use a particular verification tool, the AVISPA system. Thus, its universality may be questionable. If we wanted to use a given protocol specification in HLPSL language to test in another tool, appropriate translators should be used. Unfortunately, protocol specification is quite long, takes a few pages, and due to lack of space we do not include an example here. The example may be found in [3].

3 Examples

Below is an example of specification of one of the Kerberos protocols versions, with the use of CL.

Protocol specification:

1. A -> S : A, B,
2. S -> A : $<T, L, K, B>_{K\_AS}$, $<T, L, K, A>_{K\_BS}$,
3. A -> B : $<A, T>_{K}$, $<T, L, K, A>_{K\_BS}$,
4. B -> A : $<T>_{K}$.

Implementation of the protocol is possible assuming that users of a system use safe, encrypted communication channels, where encrypting is made with the use of the earlier stated symmetric keys shared by a server and particular users. The aim of the protocol is the distribution of a new session key between two users. Moreover, protocol has to provide a stated validity period of the key, the so-called *lifetime*. The description of protocol is as follows: in the first step user a sends to an entrusted server S willingness to start a safe communication with user B. Server generates a new symmetric key K and prepares two cryptograms encrypted by symmetric keys shared by both users. The cryptograms contain, respectively, a time ticket proving the time of generating a new key, a new key, and users' IDs. In the second step, both cryptograms are sent by a server to a user willing to start communication, that is to A. A reads the content of a cryptogram, receives a new session key K, and generates a new cryptogram encrypted with a newly received key K which includes the time ticket and ID. The cryptogram together with the previous one directed to user B, is sent to him by a in the third step. B reads the content of cryptogram from server and user A. Then, in the fourth step B confirms to a the receiving of a new key and staring mutual communication. Due to the fact that the aim of this article is not an analysis of protocol but merely of specification methods, we will not describe properties which are guaranteed by the use of the described protocol.

As an example of CAPSL the specification of NSPK protocol (Needham Schroeder Public Key Authentication Protocol) is discussed. The specification of NSPK protocol in CAPSL is as follows:

```
PROTOCOL NSPK;
VARIABLES
A, B: PKUser;
Na, Nb: Nonce, CRYPTO, FRESH;
ASSUMPTIONS
HOLDS A: B;
```

```
MESSAGES
A -> B: {A, Na}pk(B);
B -> A: {Na, Nb}pk(A);
A -> B: {Nb}pk(B);
GOALS
SECRET Na;
SECRET Nb;
PRECEDES A: B | Na;
PRECEDES B: A | Nb;
END;
```

Apparently, the specification consists of a few sections. After a header PROTOCOL NSPK there is a section of variables declaration. Variables A and B represent network users that execute the protocol. PKUser means that the users use the public key cryptography. This type is essential to state possibilities of decoding cryptogram encrypted by an appropriate key. Other variables, Na,Nb are of Nonce Type. Additionally, they are marked as CRYPTO to indicate that they are unique, that is, they cannot be guessed randomly and FRESH, which means that they are newly generated.

After the section of variables declaration there is the section of ASSUMPTIONS. In the discussed case the only assumption is indicating the initiator of a protocol, which is user A. The recipient is user B. Then, in specification there is a section of information exchange scheme during execution of protocol (external actions) in the section MESSAGES.

The actions are described in a well-known way similar to specification of CL. The abbreviation pk(A) means a public key of a user A. Secret keys are marked as sk(A). In the language, it is assumed that keys in asymmetric cryptography are constant and invariable during the protocol lasting. Na,Nb mean, respectively, quasi-random numbers of users. After the section of declaration of message exchange structure, there is a section GOALS describing the goals of the protocol.

The first two lines do not require comments. It is important to keep sent numbers Na,Nb secret. The last two lines concern mutual authentication of users A,B. In order to achieve it, CAPSL takes advantage of identity confirmation by exchanging secret numbers Na,Nb. Other lines mean that a has to receive number Na from B and vice versa.

4 ProToc Language

Now, we will show the specification language, which was prepared for the automatic verification of protocols made according to the methodology described in papers [5–10]. The language facilitates a full specification of protocol, both external actions (such as CL) and all internal actions essential in terms of verification. The introduction of four Intruder models have been foreseen in the language: dy—Dolev–Yao model [4], rdy—restriced model of Dolev–Yao, li—lazy Intruder model, rli—restricted lazy Intruder model. Adequacy of a language to test the protocol properties is confirmed by carried-out, published experiments [5–10]. The language has been prepared in two versions: one enabling modeling the time-dependent protocols (expressing time properties), and the other timeless version included in the first one. Computational structures defined in chapters [5–10] may be treated as semantics of a discussed language.

As an example of protocol specification in ProToc language the specification of NSPK protocol is shown below:

```
  users(2);
  steps(3);
  players(3);
  intruder(dy);
  protocol:
  p_1,p_2;ident_p_1,nonce_p_1,+key_p_2;nonce_p_1;
<+key_p_2,nonc e_p_1|ident_p_1>;
  p_2,p_1;nonce_p_1,nonce_p_2,+key_p_1:nonce_p_2;
<+key_p_1,nonce e_p_1|nonce_p_2>;
  p_1,p_2;nonce_p_2,+key_p_2;<+key_p_2,nonce_p_2>;
  sessions:
  (p_1, p_2);
  (p_1, p_-2);
  (p_-1, p_2);
```

The above specification may be compared to the specification in CAPSL or HLPSL.

Another example is the specification of protocol WMF as an example of protocol dependent on time.

```
  users(3);
  steps(2);
  players(4);
  intruder(rdy);
  protocol;
  p_1, p#1 ; time_1 ; ident_p_1, ident_p_2, tau_p_1,
key_p_1_2 key_p_1#1 ; tau_p_1, key_p_1_2 ; ident_p_1,
<key_p_1#1, tau_p_1 |ident_p_1|key_p_1_2>; time_1
- tau_p_1 < ltime;
  p#1, p_2 ; time_2 ; tau_p#1, ident_p_1, key_p_1_2,
key_ p_2#1 ; tau_ p#1 ;<key_p_2#1, tau_ p#1 | ident_p_1
|key_p_1_2> ; time_2 - tau_ p#1 < ltime;
  sessions:
  (p_1, p_2, p#1);
  (p_1, p_-2, p#1);
  (p_-1, p_2, p#1);
```

It may be observed that ProToc language facilitates the specification of time dependencies presumed to the proper protocol functioning.

ProToc language has finally been shown formally, that is, it has its own grammar. This facilitates the use of automatic generators such as, e.g., ANTLR or bison to create lexis and syntax analyzers. Thanks to this, the process of extending and maintaining the language is simpler—the elements of language are considered on the level of grammar rules and not of classes and functions of programming language. The grammar shown below consists of 24 rules describing particular elements of a language. Particular lexical symbols are marked with the marking literals and capital letters.

```
  protocol : header protocolSteps;
  header : numUsers numPlayers numSteps numNonce;
  numUsers : ’u’ ’=’ NUMBER SEMI;
  numPlayers : ’p’ ’=’ NUMBER SEMI;
  numSteps : ’s’ ’=’ NUMBER SEMI;
  numNonce : ’n’ ’=’ NUMBER SEMI;
  protocolSteps : PROTOCOL SEMI step+;
  step : players SEMI need? SEMI generated?
SEMI message SEMI;
  players: player COMMA player;
  player: PLAYERNR | SERVERNR;
  need: singleNeed (’,’ singleNeed)*;
  singleNeed: hash | (NONCENR | IDENT | KEY);
  generated: singleGenerated (’,’ singleGenerated)*;
  singleGenerated: (hash | (NONCENR | KEY));
  message: singleMessageContent (BAR
singleMessageContent)*;
  singleMessageContent: cryptogram | hashDef | (IDENT |
NONCENR | KEY);
  cryptogram: LT KEY COMMA cryptContent GT;
  cryptContent: singleCryptContent (BAR
singleCryptContent)*;
  singleCryptContent: cryptogram | hash | (IDENT |
NONCENR | KEY);
  hash: HASH LPAREN NUMBER RPAREN;
  hashDef: LCURLY hash COMMA hashContent RCURLY;
  hashContent: cryptContent;
  text: TEXTNR;
```

`ProToc` specification language described the implementation of cryptographic protocol. Particular language scripts consist of two parts: a section introducing basic information about the protocol `(header)` and the description of the particular steps of the protocol `(protocolSteps)`.

The introductory section introduces the number of users and players (users as well as intruders) taking part in an implementation of protocol. Moreover, the number of steps necessary for the implementation of protocol and the number of nonces for each user must be detailed. Each of the elements is saved in a similar way—the abbreviation of the name (for users it is the letter u, for players p, number of steps—s, number of nonces—n), operator of equalities, integer, and semicolon. For example:

```
  u = 2;
```

```
p = 3;
s = 3;
n = 1;
```

means the protocol with two users, three players, and three steps of implementation of protocol and one nonce for each user.

The protocol steps section begins with the keyword `protocol` after which a semicolon is put. Then, the particular steps are specified. The description of a single step of protocol (`Step` rule) assumes specification of particular players who exchange information at a given time (in grammar it is described as players rule) and who are necessary to create data (such as players' ID, keys, nonces, etc.) of exchange messages (`information` rule) which is generated during element step (`generated` rule) and exchanged message (`message` rule). Each of the mentioned step elements is separated by a semicolon.

The components of a message protocol are symbols, hashtags, and cryptograms. Since white spaces (e.g., the only possible writing of the first user is `p_1` and not `p _ 1 or p_ 1 or p _1`) must not be used as protocol symbols, the numbers of players, keys, hashtag numbers, party IDs, open text, and nonces are written as lexical symbols—respectively `PLAYERNR, KEY, HASHNR, IDENT, TEXT i NONCE`.

The most complicated component of a protocol is a key. The language permits public and symmetric keys. The key description begins with the letter K, which is followed by optional plus and minus signs—no sign means symmetric key, minus sign means secret key, plus sign means public key. To transcript public and secret keys, it is enough to specify the number of a key owner and the ordinal number of a key in parenthesis. The transcription of symmetric key requires both users. For example, : `k+_1(1)` means the first public key of the first user, `k+#1(2)` means the second public key of the first server, `k_1_2` means symmetric key of the first and the second user, `k#1_1` means a symmetric key of the first server and the first user.

Cryptogram is transcribed in angle brackets, its first component is a key which is used to prepare a cryptogram, then after a comma there are further elements which had been encrypted by this key—these are separated by a symbol of vertical line. For example, `<k+_2(1),i(p_1)|n_1(1)>` means the part of message encrypted by public key of the second user, including the ID of this user and his/her first nonce.

Hash is a single number calculated on the basis of further implemented elements.

5 Examples

The specification of the Wide-Mouth Frog (WMF) protocol is as follows:

```
u=3;
p=4;
s=2;
protocol;
```

```
  p_1,p#1;t_1(1),i(p_2),k_1_2(1),k_1#1(1);k_1_2(1);
<k_1#1(1),t_1 (1)| i(p_2)|k_1_2(1)>;
  p#1,p_2;t#1(1),i(p_1),k_1_2(1),k_2#1(1);k_1_2(1);
<k_2#1(1),t#1 (1)|i(p_1)|k_1_2(1)>;
  sessions:
  (p_1,p_2,p#1);
  (p_1,p_-2,p#1);
  (p_-1,p_2,p#1);
```

Three users and three players take part in the execution of the protocol. WMF protocol is executed in two steps. In the first part of the first step, there are symbols meaning players who communicate with one another, that is, the first player and the first server. The second part of this step contains symbols meaning objects which will be the basis of a message. There is a time stamp of the first player, ID of the second player, and two symmetric keys, shared between the first and the second player, as well as between the first player and server.

In the further part of step specification, there is a key shared between players of protocol sent in a message. The last part of the step is a cryptogram. It is created from symbols which are in the second part of this step.

The specification of the second step of WMF protocol may be read in a similar way. Another example of specification in ProToc language is the specification of the Andrew protocol:

```
  u=2;
  p=3;
  s=4;
  n=1;
  protocol;
  p_1,p_2;i(p_1),k_1_2(1),n_1(1);n_1(1);i(p_1)|
<k_1_2(1),n_1(1)>;
  p_2,p_1;n_2(1),k_1_2(1),n_1(1);n_1(1),n_2(1);
<k_1_2(1),n_1(1)| n_2(1)>;
  p_1,p_2;n_2(1),k_1_2(1);n_2(1);<k_1_2(1),n_2(1)>;
  p_2,p_1;n_2(2),k_1_2(1),k_1_2(2);k_1_2(2),n_1(2);
<k_1_2(1),k_1_2 (2)|n_2(2)>;
  sessions:
  (p_1,p_2);
  (p_1,p_-2);
  (p_-1,p_2);
```

In the execution of the protocol there are three users (two players and entrusted server). Andrew protocol is executed in four steps. In the first part of the first step there are symbols meaning players who communicate with one another, that is, the first and the second players. The second part of this step contains symbols meaning objects which will be the basis of the message. The data included are the following: ID of the first player, symmetric key shared between the first and the second players, as well as the number of first nonce of the first player. In further part of step specification, there is the first nonce of the first player sent in a message. The last part of the step is cryptogram. It is created from symbols of the second part of this step. The specification of further steps of Andrew protocol may be read similarly to WMF protocol. It should be remembered that in the last step there are nonce and a symmetric key with further numbers.

6 Summary

This article describes methods used till now as well as a new original method of security protocols specification. The simplest specification language CL and its advantages and disadvantages have been presented. Then, CAPSL and HLPSL (prepared and used within AVISPA project) languages have been shown. Also, a new and original attitude toward protocols specification, which is a new language ProToc, has been described. Grammar and examples of protocol specification have been presented. Moreover, the language facilitates protocol specification dependent on time, including time dependencies which must be followed during a proper execution of a protocol. ProToc facilitates a full specification of protocol necessary for automatic verification described in dissertations [5–10]. It facilitates testing the correctness of a protocol for four different models of Intruder. Computational structures constructed in dissertations [5–10] may be treated as semantics for ProToc language. Further development of presented solution predicts extension of expressiveness to the possibility of specification of a new verification parameter which will be the probability of breaking the encrypted key in case of use, slightly weaker cryptographic power of encrypting algorithms are used in protocols.

References

1. Needham, R., Schroeder, M.: Using encryption for authentication in large networks of computers. Commun. ACM **21**(12), 993–999 (1978)
2. Kościelny, Cz., Kurkowski, M., Srebrny, M.: Modern Cryptography Primer, p. 238. Springer, Berlin (2013). ISBN: 978-3-642-41385-8
3. Schneier, B.: Applied Cryptography: Protocols, Algorithms, and Source Code in C. Wiley, New York (1996)
4. Dolev, D., Yao, A.: On the security of public key protocols. IEEE Trans. Inf. Theory **29**(2), 198–208 (1983)
5. Kurkowski, M.: Formalne metody weryfikacji własności protokołów zabezpieczających w sieciach komputerowych, s. 208. wyd. Exit, Warszawa (2013)
6. Kurkowski, M., Penczek, W.: Verifying security protocols modelled by networks of automata. Fundamenta Informaticae **79**, 453–471 (2007)
7. Kurkowski, M., Penczek, W.: Verifying timed security protocols via translation to timed automata. Fundamenta Informaticae **93**(1–3), 245–259 (2009)
8. Kurkowski, M., Penczek, W.: Applying timed automata to model checking of security protocols. In: Wang, J. (ed.) Handbook of Finite State Based Models and Applications, pp. 223–254. Chapman and Hall/CRC Press, Boca Raton (2012)

9. Kurkowski, M., Penczek, W., Zbrzezny, A.: SAT-based verification of security protocols via translation to networks of automata. In: MoChart IV. LNAI, vol. 4428, pp. 146–165. Springer, New York (2006)
10. Kurkowski, M., Siedlecka-Lamch, O., Piech, H.: A new effective approach for modeling and verification of security protocols. In: Proceedings of CS&P'12, pp. 191–202. Humboldt University Press, Germany (2012)
11. Meadows, C.: Language generation and verification in the NRL protocol analyzer. In: Proceedings of the 1996 IEEE Computer Security Foundation Workshop IX, pp. 48–61. IEEE Computer Society Press (1996)
12. Meadows, C.: The NRL protocol analyzer: an overview. J. Log. Progr. **26**(2), 13–131 (1996)
13. Meadows, C.: Using the NRL protocol analyzer to examine protocol suites. In: Proceedings of the 1998 LICS Workshop on Formal Methods and Security Protocols. http://www.cs.bell-labs.com/who/nch/fmsp/program.html (1998)
14. Millen, J., CAPSL - Common Authentication Protocol Specification Language. http://www.mitre.org/research/capsl (1997)
15. The Common Authentication Protocol Specification Language (CAPSL) integrated protocol environment, SRI International, technical report (2001)
16. The CAPSL Integrated Protocol Environment: www.csl.sri.com/millen/
17. Armando, A., et al.: The AVISPA tool for the automated validation of internet security protocols and applications. In: Proceedings of 17th International Conference on Computer Aided Verification (CAV'05). LNCS, vol. 3576, pp. 281–285. Springer (2005)
18. Chevalier, Y., Compagna, L., Cuellar, J., Hankes Drielsma, P., Mantovani, J., Modersheim, S., Vigneron, L.: A high level protocol specification language for industrial security-sensitive protocols. In: Proceedings of SAPS'04. Austrian Computer Society (2004)

Parallelization of Block Encryption Algorithm Based on Piecewise Nonlinear Map

Dariusz Burak

Abstract In this paper, the results of parallelizing chaotic block encryption algorithm based on a piecewise nonlinear map are presented. A data dependence analysis of loops was applied in order to parallelize this algorithm. An OpenMP standard is used for presenting the parallelism of the algorithm. The efficiency measurement for a parallel program is shown.

Keywords Chaos-based encryption algorithm · Piecewise nonlinear map · Parallelization · OpenMP

1 Introduction

One of the very important features of cryptographic algorithms is a cipher speed. This feature is very significant in case of block ciphers because they have to work with large datasets. Therefore, it is important to parallelize the most time-consuming loops in order to achieve faster processing using multiprocessors or multicore machines. Nowadays, there are many descriptions of block ciphers based on various chaotic maps, for instance [1–6]. The important issue of chaotic ciphers is program implementation. Unlike parallel implementation of classical block ciphers, for instance AES [7], IDEA [8], there are only few parallel implementations of chaotic block ciphers, for instance [9, 10]. It looks like a research gap because only software or hardware implementation will show the real functional advantages and disadvantages of encryption algorithms.

Considering this fact, the main contribution of the study is developing a parallel algorithm in accordance with OpenMP standard of the cipher designed by Wei et al. [11] (called further WLZY encryption algorithm) based on the transformations of a source code written in the C language representing the sequential algorithm.

D. Burak (✉)
West Pomeranian University of Technology, 49 Zolnierska St., 71210 Szczecin, Poland
e-mail: dburak@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_21

2 The WLZY Encryption Algorithm

The WLZY encryption algorithm is a block cipher based on piecewise nonlinear map that operates in encryption mode on 64-bit input plaintext blocks with a 64-bit key.

A piecewise nonlinear map *f:I→I*, $I_i = [0, 1]$, I_i and $(i = 0, 1, \ldots, N)$ denotes the subinterval of [0, 1] as well as the length of this region, which can be presented as [12]:

$$F(\chi_{k+1}) = \begin{cases} \left(\frac{1}{I_{i+1}-I_i} + \mathrm{a}_i\right)(X) - \frac{a_i}{I_{i+1}-I_i}(X)^2 & \text{if} \quad \mathrm{x}_k \in [I_i, I_{i+1}) \\ 0 & \text{if} \quad \mathrm{x}_k = 0.5, \\ F(\chi_\mathrm{k} - 0.5) & \text{if} \quad \mathrm{x}_k \in (0.5, 1) \end{cases} \tag{1}$$

where: $\mathrm{X} = \mathrm{x_k}\text{-}\mathrm{a_i}, \mathrm{x_k} \in [0, 1], 0 = \mathrm{I}_0 < \mathrm{I}_1 < \cdots < \mathrm{I_i} < \cdots < \mathrm{I}_{\mathrm{N}+1} = 0.5$, $\mathrm{N} \geq 2, \mathrm{a_j} \in (-1, 0) \cup (0, 1), \mathrm{j} = 0, 1, \ldots, \mathrm{n}$ and

$$\sum_{\mathrm{i}=0}^{N-1} (I_{\mathrm{i}+1} - I_i)a_i = 0. \tag{2}$$

In order to improve the complexity and the period of chaotic sequence under finite-precision circumstances, the chaotic sequence is generated by coupled chaotic systems. In this case, the chaotic sequence is defined as [11]:

$$\theta = g(\chi_1(i), \mathrm{x}_2(i)) = \begin{cases} 1 & \text{if} \quad \mathrm{x}_1(i) > \mathrm{x}_2(i) \\ \text{NULL} & \text{if} \quad \mathrm{x}_1(i) = \mathrm{x}_2(i). \\ 0 & \text{if} \quad (i) < \mathrm{x}_2(i) \end{cases} \tag{3}$$

To get 64-bit output ciphertext blocks, the following steps should be carried out [11]:

1. Convert the plaintext into binary message M;
2. Divide the binary message into 64-bit plaintext blocks;
3. Generate two values that are relative to chaotic system iteration

$$\begin{cases} X_s = (K_1 \oplus \mathrm{K}_2 \oplus \mathrm{K}_3 \oplus \mathrm{K}_4 \oplus \mathrm{K}_5 \oplus \mathrm{K}_6 \oplus \mathrm{K}_7 \oplus \mathrm{K}_8)/256 \\ N_s = (K_1 + \mathrm{K}_2 + \mathrm{K}_3 + \mathrm{K}_4 + \mathrm{K}_5 + \mathrm{K}_6 + \mathrm{K}_7 + \mathrm{K}_8) \bmod 256. \end{cases} \tag{4}$$

4. Calculate the initial value and iteration times of the chaotic system based on piecewise nonlinear map

$$\begin{cases} \mathrm{X} = (X_s + \mathrm{R}_{i-1}/65535) \bmod 1 \\ \mathrm{N} = \mathrm{floor}(\mathrm{N} + \mathrm{X} \times 256), \end{cases} \tag{5}$$

where: R_{i-1}—is the right half part of the output block of previous round.

5. Iterate the piecewise nonlinear map N times with the initial value X, and obtain binary sequences $A_j = B_i^1 B_i^2 \ldots B_i^{64}$ and $A_j' = B_i^{65} B_i^{66} \ldots B_i^{70}$.
6. Convert the binary sequences A_j' into integral value D_j, then permute the message block M_j with left cyclic shift D_j bits.
7. Perform the following manipulation with sequences M_j and A_j:

$$C_j = M_j \oplus A_j. \tag{6}$$

8. If all the plaintext blocks have already been encrypted, the algorithm would terminate. Otherwise, go to step 4.

The decryption process is similar to the encryption one. In this case, step 7 is replaced with

$$M_j = C_j \oplus A_j. \tag{7}$$

A detailed description of the WLZY encryption algorithm is given in [11].

3 Parallelization Process of the WLZY Encryption Algorithm

Considering the fact that the proposed algorithm works in block manner, it is necessary to prepare a C language source code representing the sequential WLZY encryption algorithm in ECB mode of operation. The source code of such algorithm contains 29 *for* loops. Twenty of them include no I/O functions.

In order to find data dependencies in program loops, a research tool for analyzing array data dependencies called Petit was applied. Petit was developed at the University of Maryland under the Omega Project and is freely available for both DOS and UNIX systems [13, 14].

The OpenMP standard was used to present parallelized loops. The OpenMP Application Program Interface (API) [15, 16] supports multiplatform shared memory parallel programming in C/C++ and Fortran languages on all architectures, including Unix and Windows NT platforms. OpenMP is a collection of compiler directives, library routines, and environment variables which could be used to specify shared memory parallelism. OpenMP directives extend a sequential programming language with some specialized constructs: Single Program Multiple Data (SPMD) constructs, work-sharing constructs, synchronization constructs, and help to operate on both shared and private data. An OpenMP program begins execution as a single task (called a master thread). When a parallel construct is encountered, the master thread creates a team of threads. The statements within the parallel construct are executed in parallel by each thread in the team. At the end of the parallel construct, the threads of the team are synchronized. Then only the master thread continues execution until the next parallel construct is encountered. To build a valid parallel code, it is necessary to preserve all dependencies, data conflicts, and requirements regarding parallelism of a program [15, 16].

The parallelization process of the WLZY encryption algorithm consists of the following three stages:

1. carrying out the data dependence analysis of a sequential source code in order to detect parallelizable loops by using Petit program,
2. performing source code transformations,
3. constructing parallel forms of *for* loops in accordance with the OpenMP standard.

The following are the basic types of data dependencies that occur in *for* loops [17, 18]:

- a data flow dependence indicates that write-before-read ordering must be satisfied for parallel computing,
- a data antidependence indicates that read-before-write ordering should not be violated when performing computations in parallel,
- an output dependence indicates a write-before-write ordering.

Additionally, control dependence [17, 18] determines the proper ordering of running instructions.

To find the most time-consuming loops of the WLZY algorithm, experiments were carried out for an about 8 megabyte input file.

It appears that this algorithm has two computational bottlenecks: the first is enclosed in the function *wlzy_enc()* and the second in the function *wlzy_dec()*. These functions were developed to enable enciphering and deciphering the whichever number of data blocks (by analogy with functions included in the C language source code of the classic cryptographic algorithms like DES- the *des_enc()*, the *des_dec()* or IDEA- the *idea_enc()*, *idea_dec()* presented in [19]). Thus parallelization of these functions has a unique meaning.

The bodies of functions *wlzy_enc()* and *wlzy_dec()* are as follows:

```
   void wlzy_enc(wlzy_context *ctx, UINT8 *input,
 UINT8 *output, int input_length) {
   int i, nblocks=inputlen / BLOCKSIZE;
   for (i = 0; i < nblocks; i++) {
     NonlinearEncrytption(ctx, input, output);
     input+= BLOCKSIZE;
     output+= BLOCKSIZE;
   }
}.
```

```
   void wlzy_dec(wlzy_context *ctx, UINT8 *input,
 UINT8 *output, int input_length) {
   int i, nblocks=inputlen / BLOCKSIZE;
   for (i = 0; i < nblocks; i++) {
          NonlinearDecrytption(ctx, input, output);
          input+= BLOCKSIZE;
          output+= BLOCKSIZE;
       }
   }.
```

Taking into account the high degree of similarity of these functions only the first one is examined. However, the analysis is valid also in the case of the second one.

In order to apply the data dependencies analysis of the loop included in *wlzy_enc()* function, the body of the *Nonlinear_encryption()* function should be put in this loop.

Next, the following transformations should be performed to remove existing data dependencies:

1. Insert in the beginning of the loop body the following statements:
   *plaintext = &input[BLOCKSIZE*i];*
   *ciphertext = &output[BLOCKSIZE*i];*
2. Remove from the end of the loop body the following statements:
   input + = BLOCKSIZE;
   output + = BLOCKSIZE;
3. Make the privatization of the following variables:
   i, plaintext, ciphertext, X0, N, chsystem1, chsystem2, binary_sequences, Dj, pbinary, rotate, enc.

The source code of the loop transformed in accordance with the above markings is suitable to apply the following OpenMP API constructs: parallel region construct (*parallel* directive) and work-sharing construct (*for* directive).

The *wlzy_enc()* function with the parallelized most time-consuming loop is of the following form (in accordance with the OpenMP API):

```
void wlzy_enc(wlzy_context *ctx, UINT8 *input,
UINT8 *output, int input_lenght){
int i,j,nblocks,binary_sequences,Dj,enc;
float X0, N, I, ai, ai2, a, a2, chsystem1, chsystem2,
const UINT8 *plaintext;
UINT8 *ciphertext, *Achar;
UINT32 *pint, *pbinary, *rotate;
nblocks=inputlen/BLOCKSIZE;
a=0.6;
a2=0.5;
   #pragma omp parallel private (i, plaintext, ciphertext, X0,
               N, chsystem1, chsystem2, binary_sequences, Dj,
                                         pbinary, rotate, enc)
   #pragma omp for
   for (i=0; i < nblocks; i++) {
    plaintext=&input[BLOCKSIZE*i];
    ciphertext = &output[BLOCKSIZE*i];
    X0=calculate_X(ctx, Ri0);
    N=calculate_N(ctx, X0);
    interval_inicialize(N, I);
    check_ai_int(N, I, a, ai);
    check_ai_int(N, I, a2, ai2);
    chsystem1=chaos_iteraton(N, X0, I, ai2);
    chsystem2=chaos_iteraton(N, X0, I, ai);
    binary_sequences=BinarySequencesGen(chsystem1, chsystem2);
    Dj=ConvertBinary(binary_sequences, Achar);
    CharToInt(p, pint);
    pbinary=Dec2Bin(pint);
    rotate=RotateLeft(pbinary,Dj);
    enc=XOR(rotate, (UINT32*)binary_sequences);
    ConBin2Char(enc, ciphertext);
  }
}.
```

4 Experimental Results

In order to study the efficiency of the presented WLZY encryption algorithm, eight quadcore Intel Xeon Processors 7310 Series—1.60 GHz and the Intel C++ Compiler ver. 12.1 were used. The results received for an 8 megabyte input file using 2, 4, 8, 16, and 32 cores versus the only one are shown in Table 1. The number of threads is equal to the number of processors.

The total running time of the WLZY encryption algorithm consists of the following operations: data receiving from an input file, data encryption, data decryption, and data writing to an output file.

Thus the total speedup of the parallel WLZY encryption algorithm depends heavily on the four factors:

1. the degree of parallelization of the loop included in the *wlzy_enc()* function,
2. the degree of parallelization of the loop included in the *wlzy_dec()* function,
3. the method of reading data from an input file,
4. the method of writing data to an output file.

The results confirm that the loops included both the *wlzy_enc()* and the *wlzy_dec()* functions and are parallelizable with high speedup (see Table 1).

Table 1 Speedups of the parallel WLZY encryption algorithm in ECB mode

Number of processors	Number of threads	Speedup		
		Encryption	Decryption	Totality
1	1	1	1	1
2	2	1.90	1.90	1.30
4	4	3.60	3.70	1.80
8	8	5.80	6.00	2.20
16	16	5.90	6.10	2.25
32	32	5.80	5.90	2.15

The block method of reading data from an input file and writing data to an output file was used: *fread()* function and 1,024-bytes block for data reading and *fwrite()* function and 128-bytes block for data writing.

5 Conclusions

In this paper, the parallelization process of the WLZY encryption algorithm was described. The experiments have shown that the time-consuming *for* loops included in the functions responsible for the data encryption and data decryption processes are parallelizable. Therefore, the application of the parallel WLZY encryption algorithm for multiprocessor and multicore machines would considerably boost the time of the data encryption and decryption. I believe that the speedups received for these operations are satisfactory. Moreover, the developed parallel WLZY encryption algorithm can also be helpful for hardware implementations and GPU-based implementations.

References

1. Habutsu, T., Nishio, Y., Sasase, I., Mori, S.: A secret key cryptosystem using a chaotic map. IEICE Trans. Jpn. **E73**(7), l041–1044 (1990)
2. Kocarev, L., Jakimoski, G.: Logistic map as a block encryption algorithm. Phys. Lett. A **289**(4–5), 199–206 (2001)
3. Yi, X., Tan, C.H., Siew, C.K.: A new block cipher based on chaotic tent maps. IEEE Trans. Circuits Syst. I: Fundam. Theory Appl. **49**(12), 1826–1829 (2002)
4. Lian, S., Sun, J., Wang, Z.: A block cipher based on a suitable use of the chaotic standard map. Chaos, Solitons and Fractals **26**(1), 117–129 (2005)
5. Lian, S., Chen, X.: Traceable content protection based on chaos and neural networks. Appl. Soft Comput. **11**(7), 4293–4301 (2011)
6. Pejaś, J., Skrobek, A.: Chaos-based information security. In: Stavroulakis, P., Stamp, M. (eds.) Handbook of Information and Communication Security, pp. 91–128. Springer, Berlin (2010)
7. Bielecki, W., Burak, D.: Exploiting loop-level parallelism in the AES algorithm. WSEAS Trans. Comput. **5**(1), 125–133 (2006)
8. Bielecki, W., Burak, D.: Lecture notes in computer science. Parallelization of the IDEA Algorithm, pp. 635–638. Springer, Berlin (2004)
9. Burak, D., Chudzik, M.: Parallelization of the discrete chaotic block encryption algorithm. In: Wyrzykowski, R. (ed.) PPAM 2011, Part II, LNCS 7204, pp. 323–332. Springer, Berlin (2012)
10. Burak, D.: Parallelization of encryption algorithm based on chaos system and neural networks. In: Wyrzykowski, R. (ed.) PPAM 2013, Part II, LNCS 8385, pp. 364–373. Springer, Berlin (2014)
11. Wei, P., Liao, X., Steinke, T., Zhang, W., Yang, H.: A chaotic block encryption scheme based on piecewise nonlinear map. Wuhan University J. Nat. Sci. **11**(6), 1521–1524 (2006)
12. Tao, S., Ruli, W., Yixun, Y.: Generating binary bernoulli sequences based on a class of even-symmetric chaotic maps. IEEE Trans. Commun. **49**(8), 620–623 (2001)
13. Kelly, W., Maslov, V., Pugh, W., Rosser, E., Shpeisman, T., Wonnacott, D.: new user interface for petit and other extensions. User guide, (1996)
14. The Omega Project: Frameworks and algorithms for the analysis and transformation of scientific programs, http://www.cs.umd.edu/projects/omega/
15. OpenMP application program interface version 3.1, (July 2011)

16. An API for multi-platform shared-memory parallel programming in C/C++ and Fortran, http://www.openmp.org/
17. Allen, R., Kennedy, K.: Optimizing compilers for modern architectures: a dependencebased approach. Morgan Kaufmann Publishers Inc. (2001)
18. Aho, A., Lam, M., Sethi, R., Ullman, J.: Compilers: Principles, Techniques, and Tools, 2nd edn. Prentice Hall (2006)
19. Schneier, B.: Applied Cryptography: Protocols, Algorithms, and Source Code in C., 2nd edn. Wiley, New York (1995)

Statistical Properties Analysis of Key Schedule Modification in Block Cipher PP-1

Michał Apolinarski

Abstract An important element of the key schedule in block ciphers is that the generated sequences of bits representing the round keys should be independent, because it affects the quality of the cipher cryptanalysis, which is difficult for independent keys. In this article is presented the results of statistical tests carried out using NIST 800-22 statistical suite on bit sequences produced by different variants of scalable PP-1 block cipher key schedule. Motivation for this research was to simplify and accelerate PP-1 key schedule without losing the statistical quality of generated keys.

Keywords Key schedule · Round keys · Block cipher · Nist 800-22 · Statistical tests.

1 Introduction

Generating independent round keys in block ciphers is an important property of key schedule algorithms, because it affects the quality of the cipher cryptanalysis, which is difficult for independent keys [1–7]. In this article is presented the results of chosen NIST 800-220 statistical tests performed on bit sequences generated by different variants of PP-1 block cipher key schedule. Motivation for this research was to find modifications that would simplify and accelerate the PP-1 key schedule without losing the statistical quality of the generated round keys. Modifications to the key schedule consisted, among others, of conversion or removal of selected operations and examining whether generated by a modified algorithm round keys will preserve their randomness properties. The criterion of randomness will be taken as the main criterion for evaluation of new key schedule taking into account the trade-off between security and speed of round key generation.

M. Apolinarski (✉)
Institute of Control and Information Engineering, Poznan University of Technology,
Pl. M. Skłodowskiej Curie 5, 60-965 Poznań, Poland
e-mail: michal.apolinarski@put.poznan.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_22

2 Testing Method

NIST 800-22 allows to evaluate the quality of algorithms for generating pseudorandom sequences, examining how the generated bit sequence is different from the random sample. In order to carry out all statistical tests available in the package, the NIST 800-22 algorithm must generate bit sequences of length samples of $n > 10^6$. The block cipher PP-1 key schedule with 64-bit master key generates in single-run 22 round keys with a length of 64 bits, which gives sample length of 1,408 bits (concatenation of 22 round keys are treated as a single sequence samples for NIST 800-22). For samples of this length can be carried out 7 of 15 NIST 800-22 test [8] listed below:

- Frequency Test—checks the frequency of occurrences of 1 and 0 bits in the sequence and checks whether they correspond to the random sequence (recommended length of the sample $n \geq 100$).
- Block Frequency Test—counts the frequencies of the different m-bit sequences of the test block and verifies whether they appear at the same frequency (recommended length of the sample n $\geq$ 100).
- Cumulative Sums Test—checks whether the total bits (where 1 bit is equal to 1, and bit 0 is equal to -1) in particular the beams are not too large or too small, which would mean too great a number 0 or 1 in the different parts of the query sequence (recommended length of the sample $n \geq 100$).
- Runs Test—counts sequences of ones and zeros of different lengths and checks whether these numbers correspond to the random sequence (recommended length of the sample $n \geq 100$).
- Spectral DFT Test—checks whether the test sequence does not appear in periodic patterns (recommended length of the sample $n \geq 1{,}000$).
- Approximate Entropy Test—compares the frequency of overlapping blocks of length m and $m + 1$, checks if any of the blocks does not occur too often (recommended length of sample $m < \lfloor \log_2 n \rfloor - 2$).
- Parameterized Serial Test—checks whether the number of m-bit overlapping blocks is suitable (recommended length of sample $m < \lfloor \log_2 n \rfloor - 2$).

The result of each test must be greater than the acceptance threshold to be considered as sequence with good statistical properties. For all tests the threshold of acceptance was 0.980561 and the level of significance was 0.1. The results should be interpreted as the percentage of samples that meet the test. Test results that did not exceed the acceptance threshold determined in the article are underlined. All test samples were generated by the same set of master keys in each variant of the PP-1 key schedule.

3 Variants of Tested Key Schedules

This section presents the original PP-1 key schedule algorithm and proposed and tested modification versions of the key schedule in the PP-1 block cipher [9–11].

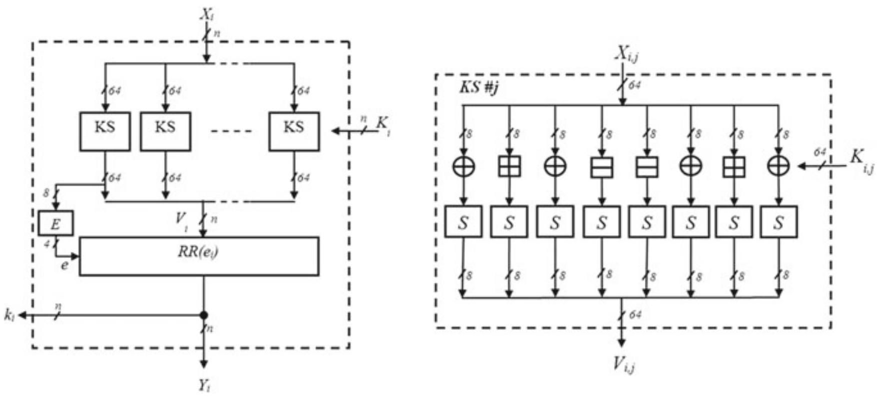


Fig. 1 Original PP-1 key schedule without any modification

3.1 Original PP-1 Key Schedule

PP-1 block cipher key schedule with n-bit master key generate 22 n-bits round keys. The round keys are generated in $2r + 1$ iterations, where r is the number of rounds (first iteration of key schedule does not produce round keys, so $k_1, k_2, \ldots, k_{2r}$ are round keys).

Figure 1 shows key schedule of scalable PP-1 block cipher. Input X_i for iteration #0 is n-bit constant: $\mathrm{B} = \mathrm{B}_1||\mathrm{B}_2||\cdots||\mathrm{B}_t$, where $\mathrm{B}_1 = 0 \times 91\mathrm{B}4769\mathrm{B}2496\mathrm{E}7\mathrm{C}$, $\mathrm{B}_j = Prm(\mathrm{B}_{j-1})$ for $j = 2, 3, \ldots, t$, where Prm is auxiliary permutation described in [9]. Input K_i for iteration #0 and #1 is computed depending on the master key length: if length of master key is equal to n, then $K_0 = k$ and $K_1 = 0n$ (concatenation of zeros), if length of master key is equal to $2n$, then key is divided into two parts k_H and k_L, giving $K_0 = k_H$ and $K_1 = k_L$. Value K_i for iterations #2 is: $K_2 = RL(B \oplus (A(K_0 \oplus K_1)))$, where $^\wedge$ means Boolean AND operation, and RL is left rotation by 1-bit, value A depends on master key length, if master key is equal to n, then $A = 0n$, if master key is equal $2n$, then $A = 1n$. Value K_i for iteration #3$\cdots$#2r is computed as $K_i = RL(K_{i-1})$. The rest of the key schedule components are:

- *KS*—main element consisting of S-block, XOR, add, sum mod 256 performed on 8-bit values, derived from 64-bit input from n-bit block X_i;
- $RR(e_i)$—right rotation by e_i bits of n-bit V_i block;
- E—component that computes 4-bit value $e_i = E(b_1, b_2, \ldots, b_n) = (b_1 \oplus b_8)(b_2 \oplus b_{10})(b_3 \oplus b_{11})(b_4 \oplus b_{12})$, based on 8-bit input, which is a concatenation of four most significant bit outputs of two leftmost S-boxes in *KS* element.

Each of the seven statistical tests that could be executed on the 1,408-bit sample was positive:

- frequency: 0.9890;
- block frequency ($m = 4$): 0.9970;

- cumulative sum: 0.9910 and 0.9900;
- runs: 0.9860;
- spectral DFT: 0.9890;
- entropy ($m = 4$): 0.9940;
- serial ($m = 4$): 0.9910 and 0.9890;

Below are the number of zeros and ones in the 1,408 generated bit sequences of the first two and the last two primary keys.

- BITSREAD = 1,408 0 s = 690 1 s = 718
- BITSREAD = 1,408 0 s = 712 1 s = 696
- $\cdots$
- BITSREAD = 1,408 0 s = 732 1 s = 676
- BITSREAD = 1,408 0 s = 744 1 s = 664

3.2 Modified PP-1 Key Schedules

This section presents the different version of PP-1 key schedule modifications for which the statistical tests were carried out.

3.2.1 Key Schedule Without Function $RR(e_i)$

Figure 2 presents the modified algorithm with removed $RR(e_i)$ component. KS element remained unchanged. All possible tests for sequence of 1,408 bits were positive:

- frequency: 0.9890;
- block frequency ($m = 4$): 0.9880;
- cumulative sum: 0.9910 and 0.9900;
- runs: 0.9840;
- spectral DFT: 0.9850;
- entropy ($m = 4$): 0.9960;
- serial ($m = 4$): 0.9960 and 0.9950;

3.2.2 Key Schedule with ADD and SUB Operations Replaced by XOR Operation

Figure 3 presents the modified algorithm wherein KS element in all arithmetic operation with input $K_{i,j}$ were replaced by XOR operation. All the tests that could be performed on a sample with a length of 1,408 bits were positive:

- frequency: 0.9890;
- block frequency ($m = 4$): 0.9940;

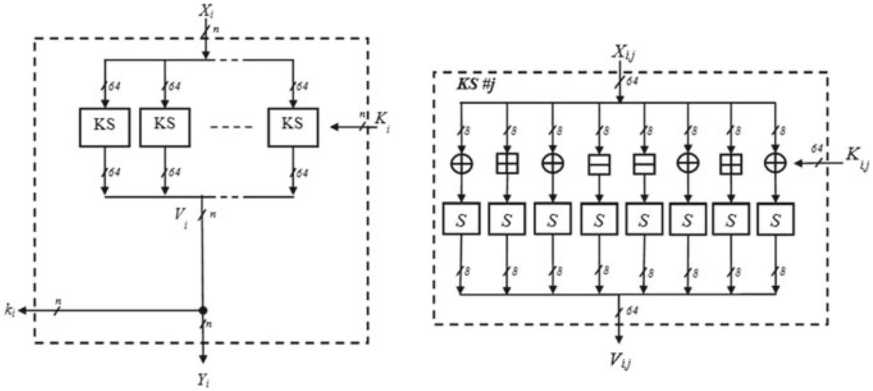


Fig. 2 Key schedule without $RR(e_i)$ function

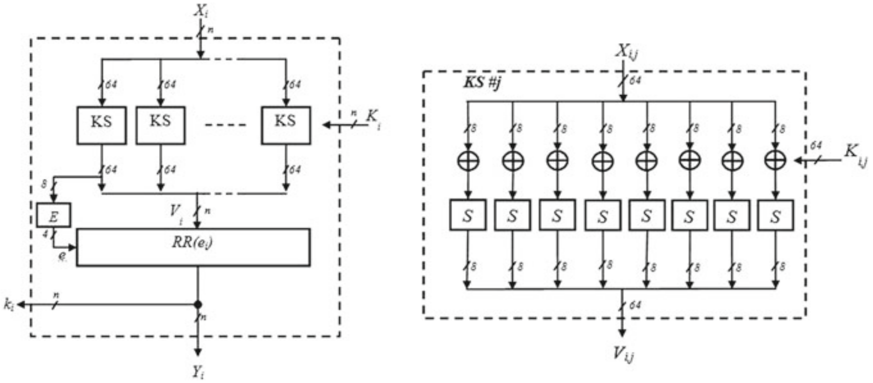


Fig. 3 Key schedule with ADD and SUB operations replaced by XOR operation

- cumulative sum: 0.9930 and 0.9880;
- runs: 0.9880;
- spectral DFT: 0.9880;
- entropy ($m = 4$): 0.9920;
- serial ($m = 4$): 0.9950 and 0.9800;

3.2.3 Key Schedule with Reduced Number of S-Boxes

Figure 4 shows the modified version of algorithm without S-boxes. Figures 5 and 6 shows key schedule with reduced number of S-boxes.

Key schedule version without S-boxes passed five of seven tests, the test of entropy: and test of series with parameter gave a negative result:

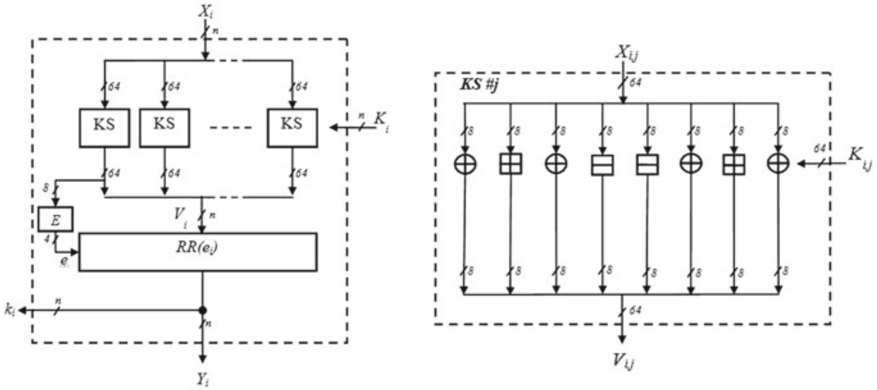


Fig. 4 Key schedule without S-boxes

- frequency: 0.9970;
- block frequency ($m = 4$): 0.9980;
- cumulative sum: 0.9960 and 0.9960;
- runs: 0.9960;
- spectral DFT: 0.9860;
- entropy ($m = 4$): $\underline{0.9740}$;
- serial ($m = 4$): 0.9870 and $\underline{0.9730}$;

The same tests were repeated for the version with four S-boxes on the transforming path number 1, 3, 5, 7 data byte (Fig. 5), the test results were also negative:

- frequency: 0.9960;
- block frequency ($m = 4$): 0.9970;
- cumulative sum: 0.9970 and 0.9940;
- runs: $\underline{0.9740}$;
- spectral DFT: 0.9860;
- entropy ($m = 4$): 0.9810;
- serial ($m = 4$): $\underline{0.9740}$ and 0.9860;

Figure 6 shows a version of the *KS* element with six S-boxes. All statistical tests results were positive:

- frequency: 0.9960;
- block frequency ($m = 4$): 0.9990;
- cumulative sum: 0.9960 and 0.9980;
- runs: 0.9950;
- spectral DFT: 0.9860;
- entropy ($m = 4$): 0.9910;
- serial ($m = 4$): 0.9890 and 0.9880;

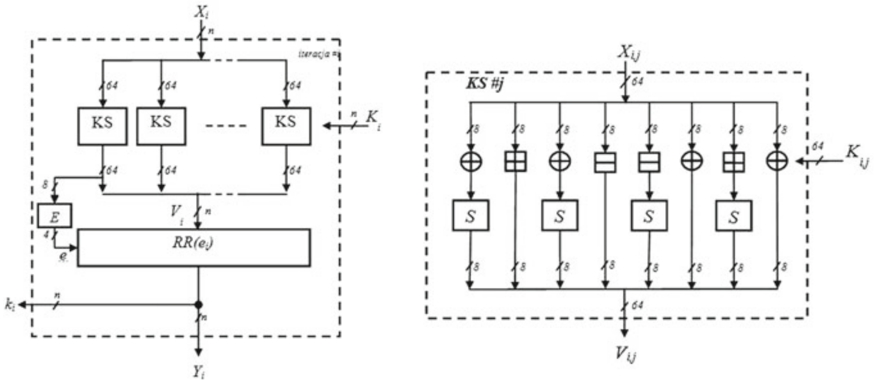


Fig. 5 Key schedule with four S-boxes

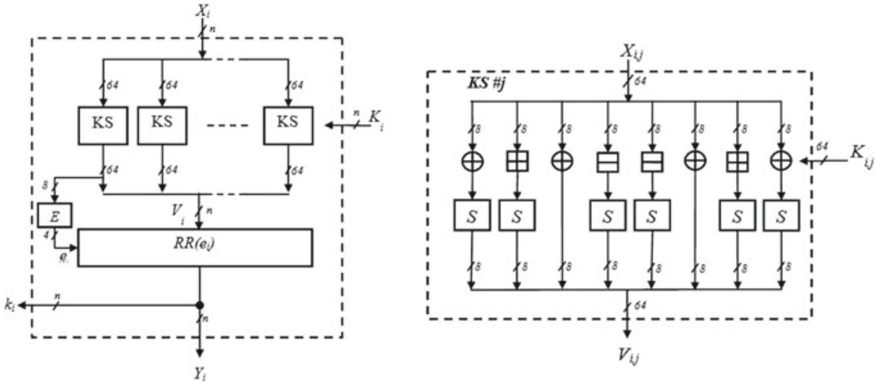


Fig. 6 Key schedule with six S-boxes

3.2.4 Key Schedule with Modified B Constant

In this variant of the key schedule the input X_i was modified by changing the original constant $B_1 = 0 \times 91B4769B2496E7C$ to the value $B_1 = 0 \times 010101010101010$, other elements of the algorithms remained unchanged. The results of four tests were negative:

- frequency: 0.9450;
- block frequency ($m = 4$): 0.9920;
- cumulative sum: 0.9380 and 0.9380;
- runs: 0.9950;
- spectral DFT: 0.9810;
- entropy ($m = 4$): 0.8770;
- serial ($m = 4$): 0.9160 and 0.9620;

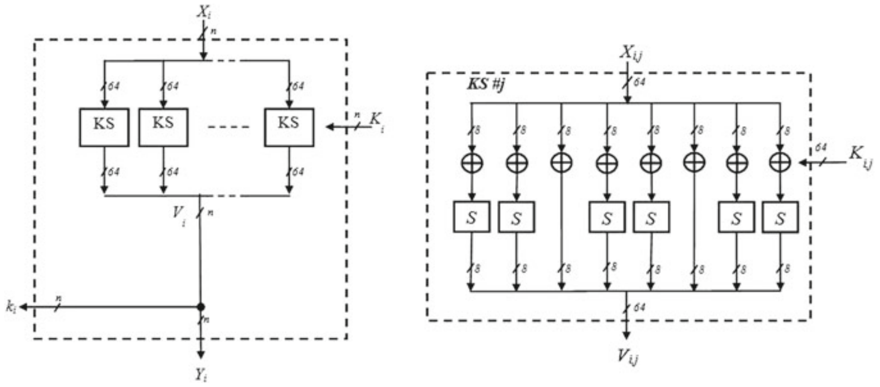


Fig. 7 Key schedule with mixed positive modifications

The tests were repeated for a constant value of $B_1 = 0 \times 0F0F0F0F0F0F0$, the results of two tests of seven were negative:

- frequency: 0.9960;
- block frequency ($m = 4$): 0.8940;
- cumulative sum: 0.9970 and 0.9970;
- runs: 0.9960;
- spectral DFT: 0.9850;
- entropy ($m = 4$): 0.9580;
- serial ($m = 4$): 0.9920 and 0.9870;

3.2.5 Key Schedule with Combination of Selected Modifications

Presented in the following section algorithm is a combination of modification, which does not cause deterioration of statistical properties. Function $RR(e_i)$ was removed, only six S-boxes were used, and all arithmetic operations were replaced by XOR operation. All tests for this new algorithm were positive:

- frequency: 0.9970;
- block frequency ($m = 4$): 0.9960;
- cumulative sum: 0.9990 and 0.9970;
- runs: 0.9880;
- spectral DFT: 0.9840;
- entropy ($m = 4$): 0.9900;
- serial ($m = 4$): 0.9900 and 0.9920;

The following shows the number of zeros and ones in the generated 1,408-bit sequence in the first two and the last two master keys.

- BITSREAD = 1,408 0 s = 718 1 s = 690

- BITSREAD = 1,408 0 s = 715 1 s = 693
- · · ·
- BITSREAD = 1,408 0 s = 721 1 s = 687
- BITSREAD = 1,408 0 s = 719 1 s = 689

3.2.6 Key Schedule with Combination of Selected Modifications and Global Rotation by 8 Bits

Figure 8 shows the same algorithm as in the previous version, but also taking into account the global rotation by fixed 8-bit value to the right. Results were negative:

- frequency: 0.9940;
- block frequency ($m = 4$): 0.9940;
- cumulative sum: 0.9900 and 0.9850;
- runs: 0.9960;
- spectral DFT: 0.9860;
- entropy ($m = 4$): 0.9890;
- serial ($m = 4$): 0.9870 and 0.9760;

The below list presents the number of zeros and ones in the generated 1,408-bit sequence from in the first two and the last two master keys in this variant of key schedule

- BITSREAD = 1,408 0 s = 695 1 s = 713
- BITSREAD = 1,408 0 s = 669 1 s = 739
- · · ·
- BITSREAD = 1,408 0 s = 705 1 s = 703
- BITSREAD = 1,408 0 s = 705 1 s = 703

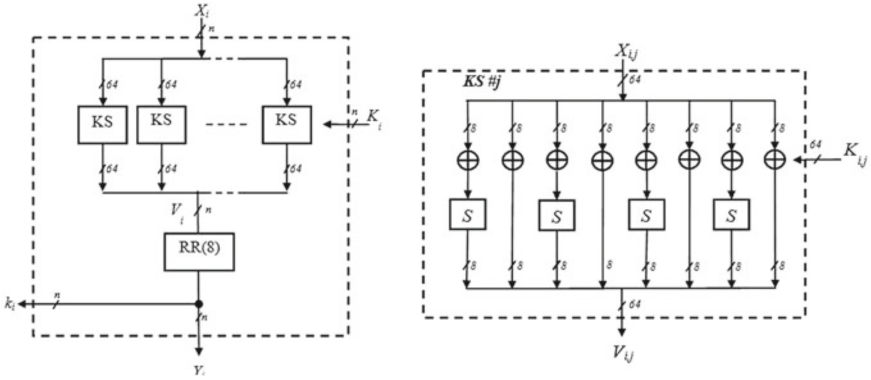


Fig. 8 Key schedule from Fig. 7 with additional rotation by 8 bits

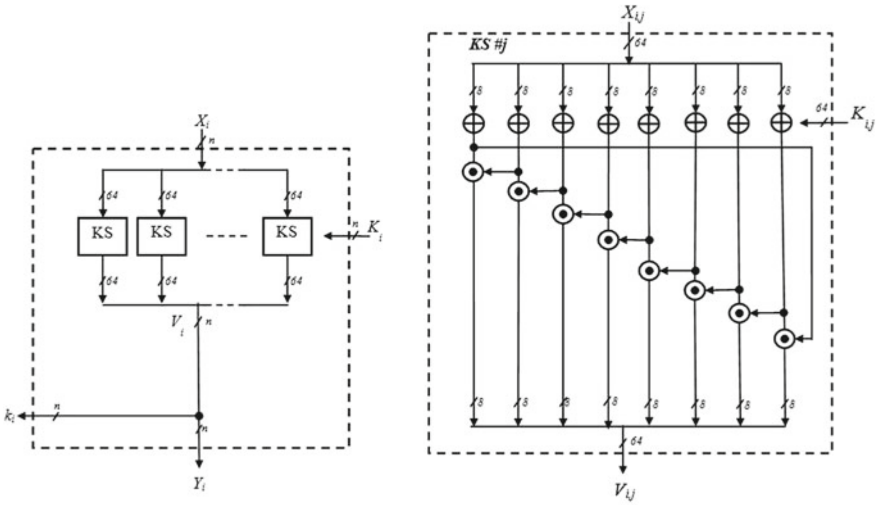


Fig. 9 Key schedule using multiplication

3.2.7 Key Schedule with Multiplication

Figure 9 shows element *KS* of key schedule algorithm that uses the XOR operation on the input $X_{i,j}$ and $K_{i,j}$, then the multiplication is performed *ab* mod 257, assuming that 256 is replaced by 0 (circle with dot symbol). Results of three tests were negative:

- frequency: 0.9860;
- block frequency ($m = 4$): 0.9900;
- cumulative sum: 0.9830 and 0.9850;
- runs: <u>0.9790</u>;
- spectral DFT: 0.9870;
- entropy ($m = 4$): <u>0.9790</u>;
- serial ($m = 4$): <u>0.9680</u> and 0.9800;

4 Comparison of Key Schedule Execution Time

This section presents the key schedule execution time using different versions of key schedule algorithm presented in this article. Due to the specific implementation aimed at generating samples for testing NIST 800-22 measured time takes into account the whole process of generating the sample (i.e., the duration of generating 1,000 samples for 1,408 each and preparing input data as binary file)—so important is the difference between times and not pure time of generating. Presented times are the average time of five executions:

- original algorithm (Fig. 1): 2.091 s
- version without function $RR(e_i)$ (Fig. 2): 2.057 s

- version with modified B constant: 2.091 s
- version with ADD and SUB operations replaced by XOR (Fig. 3): 2.075 s
- version without S-boxes (Fig. 4): 2.041 s
- version with four S-boxes (Fig. 5): 2.061 s
- version with six S-boxes (Fig. 6): 2.074 s
- version with combination of positive modifications (Fig. 7): 2.028 s
- version with modifications and fixed rotation (Fig. 8): 2.061 s
- version with multiplication (Fig. 9): 2.019 s

5 Conclusions

This research shows that it is possible to reduce the original algorithm to generate round keys for PP-1 block cipher to figures presented in Fig. 6 without compromising the quality of generated round keys taking as a criterion for assessing the quality of the NIST 800-22 statistical tests. In that version all possible to carry out tests on samples with length of 1,408 bits were successful. The research also showed the importance of using a good quality B constant, which enables to input to the key schedule good bit string even if the master key is poor. Another issue worth testing and analysis are operations that prepare input for iteration #0 (which does not produce the key) and iteration #1 and #2, because these operations depend largely on the quality of the generated keys. Further research should be aimed at developing methods of designing and testing block cipher's key schedule algorithms that produce good round keys.

Acknowledgments This research has been supported by the Polish Ministry of Science and Higher Education under grant 04/45/DSPB/0122.

References

1. Biham E.: New types of cryptanalytic attacks using related keys. Workshop on the theory and application of cryptographic techniques on Advances in cryptology, p. 398–409 (Lofthus, Norway, January 1994)
2. Biham E., Dunkelman O., Keller N.: Related-key Boomerang and rectangle attacks. In: Proceedings of the 24th annual international conference on theory and applications of cryptographic techniques, pp. 22–26 (Aarhus, Denmark, May 2005)
3. Biham E., Dunkelman O., Keller N.: A unified approach to related-key attacks. In: Fast Software Encryption: 15th International Workshop, FSE 2008, Lausanne, Switzerland pp. 10–13, Revised Selected Papers (Springer, Berlin, February 2008)
4. Biham, E., Shamir, A.: Differential Cryptanalysis of the Data Encryption Standard. Springer, New York (1993)
5. Biryukov A., Khovratovich D.: Related-key cryptanalysis of the full AES-192 and AES-256. In: Proceedings of the 15th international conference on the theory and application of cryptology and information security: advances in cryptology, Tokyo, Japan, pp. 06–10 (December 2009)

6. Biryukov A., Khovratovich D., Nikolić I.: distinguisher and related-key attack on the full AES-256. In: Proceedings of the 29th annual international cryptology conference on advances in cryptology, Santa Barbara, CA, pp. 16–20 (August 2009)
7. Kim J, Hong S., Preneel B.:" Related-key rectangle attacks on reduced AES-192 and AES-256. In: Proceedings of the 14th international conference on fast software encryption, Luxembourg, pp. 26–28 (March 2007)
8. Rukhin, A.:A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST Special Publication 800–22, revision 2 (2008)
9. Bucholc, K., Chmiel, K., Grocholewska-Czuryło, A., Idzikowska, E., Janicka-Lipska, I., Stokłosa, J.: Scalable PP-1 block cipher. Int. J. Appl. Math. Comput. Sci. **20**(2), 401–411 (2010)
10. Chmiel, K., Grocholewska-Czurylo, A., Socha, P., Stoklosa, J.: Scalable cipher for limited resources. Pol. J. Environ. Stud. **17**(4C), 371–377 (2008)
11. Chmiel, K., Grocholewska-Czurylo, A., Stoklosa, J.: Involutional block cipher for limited resources. IEEE GLOBECOM **1**, 1852–1856 (2008)

Fault Detection in PP-2 Symmetric Block Cipher

Ewa Idzikowska

Abstract Concurrent Error Detection (CED) techniques based on hardware or time redundancy are widely used to enhance system dependability and to detect fault injection attacks. In this paper, we present two approaches to countermeasure against fault injections into a symmetric block cipher PP-2. We proposed the hardware redundancy CED technique based on parity checks and time redundancy technique based on an inverse relationship between encryption and decryption at the round level of PP-2. Simulation results for single and multiple as well as transient and permanent faults are presented.

Keywords CED · PP-2 · Fault detection · Parity checks · Time redundancy

1 Introduction

Hardware implementations of encryption algorithms leak information via side channels such as power consumed by the operations, measurement of time, or deliberate introduction of faults. Mathematical analysis can be combined with the side channel information to reveal the secret key. These side channel analysis attacks are much more powerful compared to attacks based on mathematical analysis.

Boneh, DeMillo, and Lipton introduced fault-based side channel attacks based on the observation that errors induced in the hardware devices leak information about the implemented encryption algorithm. These techniques have been increasingly studied after the publication in 1997 [1].

There are different types of faults and methods of fault injection in encryption algorithms. The faults can be transient or permanent. Several transient and permanent faults and methods of fault injection such as varying supply voltage, external clocks, temperature, or induced faults using white light, laser, and X-ray methods of fault injection are discussed in detail in [2].

E. Idzikowska (✉)
Poznań University of Technology, pl. M. Skłodowskiej-Curie 5, Poznań, Poland
e-mail: ewa.idzikowska@put.poznan.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_23

Countermeasures against fault attacks can be deployed in hardware or software and generally help to detect faults. In practice, most proposed schemes are based on classical error-detecting techniques using hardware or time redundancies [3, 4].

In this paper we present two approaches to CED in hardware implementations of symmetric block cipher. One of these CED techniques is based on the parity bits, the other exploits inverse relationships that exist between encryption and decryption. Any input data that is passed successively through encryption and decryption algorithm is recovered.

We will analyze the detection of errors in PP-2 block cipher [5]. PP-2 is considered for use in essential security services and Concurrent Error Detection (CED) is very important. The design goal is to achieve 100 % error detection with minimal penalty.

This paper is organized as follows. In Sect. 2 we present the PP-2 symmetric block cipher. In Sect. 3 error detection techniques are described. Simulation results are presented in Sect. 4. Section 5 concludes the paper.

2 The PP-2 Cipher

The scalable PP-2 cipher is a symmetric block cipher designed at the Institute of Control and Information Engineering, Poznań University of Technology. Symmetric block ciphers have an iterative looping structure. All the rounds of encryption and decryption are identical in general, with each round using several operations and round key(s) to process the input data. The PP-2 cipher in r rounds processes data blocks of n bits, using cipher keys with lengths of n. It is described in detail in [5].

The PP-2 algorithm is an SP-network. One round of the algorithm is presented in Fig. 1. It consists of $t = n/64$ parallel processing paths. A 64-bit nonlinear operation is performed in each path. The 64-bit block is processed as 8-bit subblocks by four types of transformations: 8×8 S-boxes, XOR, addition, and subtraction *modulo* 256 of integers, represented by respective bytes. Additionally, an n bit permutation P is used.

Encryption and decryption round is shown in Fig. 2. Functions P, SL, KL and round keys $k_1, \ldots, k_i, \ldots, k_r$ are used in the encryption process. In the decryption process inverse functions P^{-1}, SL^{-1} and KL^{-1} are used. Round keys must be used in the reverse order, i.e., $k_r, \ldots, k_i, \ldots, k_1$.

Symmetric block ciphers have an iterative looping structure. All the rounds of encryption and decryption are identical in general, with each round using several operations and round keys to process the input data. Decryption is the inverse of encryption.

A round of encryption (decryption) performs a series of operations on the input data block and the round keys to generate the intermediate output data block. An output data block is then used as input data block for the next round. After a predetermined number of rounds, cipher (plain) text is generated.

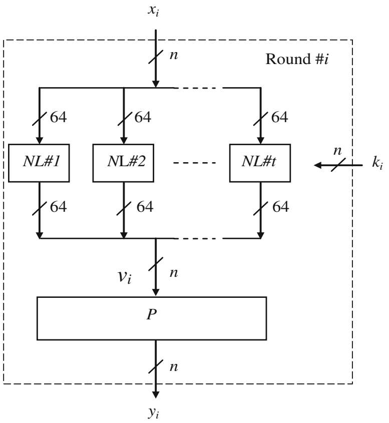


Fig. 1 One round of PP-2 ($i = 1, 2, \ldots, r$) [5]

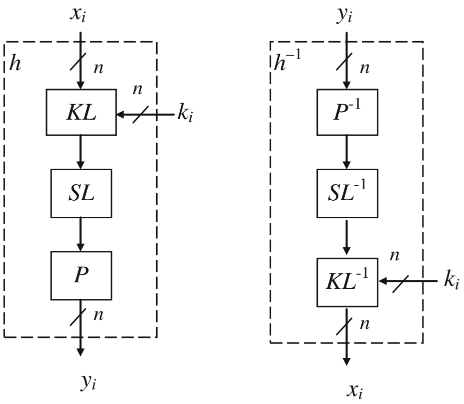


Fig. 2 Encryption and decryption round #i of PP-2 cipher ($i = 1, 2, \ldots, r$) [5]

The inverse relationship in PP-2 exists at three levels—operations, rounds, and algorithm. At the operation level, encryption and decryption use mutually inverse operations, such as addition-subtraction, left rotation-right rotation, XOR-XOR, etc. At the round level, the sequence of operations in the round of decryption is the reverse of the sequence of operations in the round of encryption. At the algorithm level, the order of rounds in decryption is the reverse of that in encryption (Fig. 2).

3 Error Detection Techniques

In this section we describe two solutions for the low cost concurrent error detection in substitution-permutation network of PP-2 cipher. One based on space redundancy and the other based on time redundancy.

3.1 Parity Checks

The round of an unprotected block cipher implementation is shown in Fig. 1, *NL* represents nonlinear element of PP-2 and is shown in Fig. 3. It consists of a linear *KL* element and a nonlinear *SL* element, (representing nonlinear substitution boxes). The basic purpose of the countermeasure is to add parity bits to the scheme in order to detect errors during the execution of the encryption and decryption algorithm.

The 64-bit input block is processed as 8-bit subblocks (Fig. 3). One of these subblocks with parity bits is shown in Fig. 4. Input parity $\mathrm{P}(x_{ik})$ is computed for each such 8-bit subblock. The parity of 64-bit input consists of 8 bits. The linear element *KL* does not involve any modification of the previously defined parity, which means that $\mathrm{P}(x_{ik}) = \mathrm{P}(z_{ik})$. If not, a fault is detected. Multiple faults of even order will not be detected by such scheme if they are located in one subblock. $\mathrm{P}(x_{ik})$ and $\mathrm{P}(z_{ik})$ can be determined by XOR gates.

An S-box is a nonlinear element usually implemented as a 256 × 8 bits memory. It consists of a data storage section and an address decoding circuit. Two additional 256 bit vectors are used to increase the dependability and detect input, output and internal memory errors of the S-box. One of the vectors contains parity bits predicted for all input data bytes, the other contains parity bits predicted for outgoing data (Fig. 4). Thus the solution demands only 512 additional memory bits (redundancy for one S-box equal to 25 %) and simple combinational circuit, and guarantees quite good fault coverage.

Permutation *P* preserves parity as *KL*. The parity of 64-bit input of *P* must be the same as the parity of 64-bit output of *P*.

Capability of single and multiple, transient and permanent fault detection using this scheme of parity prediction is presented in Sect. 4.

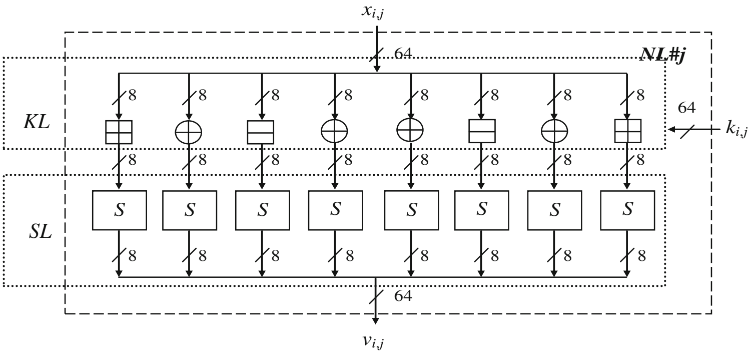


Fig. 3 Nonlinear element *NL* of PP-2 cipher ($j = 1, 2, \ldots, t$)

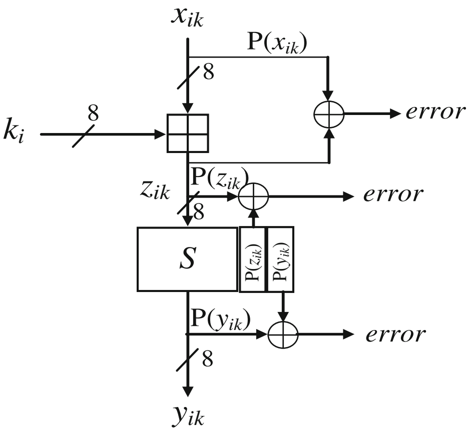


Fig. 4 A subblock of element *NL* with parity check ($k = 1, 2, \ldots, 8$, $i = 1, 2, \ldots, r$)

3.2 Time Redundancy

In symmetric block ciphers any input data that is passed successively through one encryption round, in the corresponding decryption round is recovered.

An encryption device usually consists of an encryption module, a decryption module, and RAM to store keys. Since a symmetric encryption algorithm uses the same set of round keys for both encryption and decryption, they can be generated, stored in the key RAM, and retrieved in any order depending on whether encryption or decryption is in progress. We assume that either encryption or decryption is performed at a time. This means that the other module is idle and can be used for CED.

In most symmetric block cipher algorithms, the first round of encryption corresponds to the last round of decryption, the second round of encryption corresponds to the second last round of decryption, and so on. Based on this observation, fault detection computations can be performed at the round level. At the beginning of each encryption round input data is stored in a register, before being fed to the round module. After an encryption round is finished, its output is fed to the corresponding decryption round. Output of the decryption round is then compared with the input data that was saved in the register. If there is a mismatch, encryption process is halted (Fig. 5).

Performance penalty for encrypting one block of data the round level CED is only a round, because the current round of decryption (for CED) can start concurrently with the next round of encryption.

The fault detection latency is twice the time required for one round. Hardware redundancy is due to the additional *n* bit registers, comparators, multiplexers, and controller.

In Sect. 3.1 faults are detected after operations *KL, SL, P*, and can be indicated immediately, on the operation level, or after the round execution.

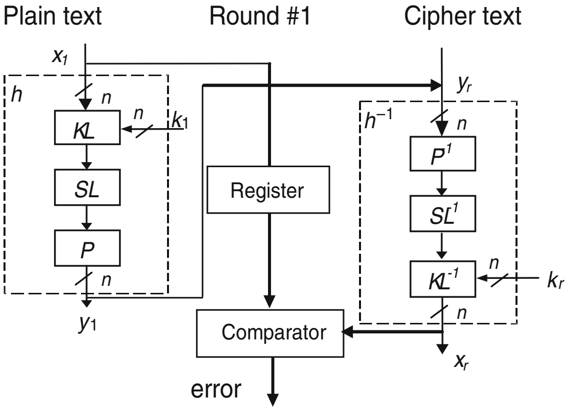


Fig. 5 One round encryption with round level CED [3]

Each encryption (decryption) round can be partitioned into operations such that the operations of encryption and corresponding operations of decryption satisfy the inverse relationship. Passing the input data through an operation in the encryption round and the corresponding inverse-operation in decryption round we recover the original input data. Such an operation level CED improves the fault detection latency and in addition it localizes the fault to the hardware implementing the operation. On the other hand, complexity of the design increases. Therefore, we take into consideration only round level error detection in both cases.

Capability of single and multiple, transient, and permanent fault detection using these CED schemes is presented in Sect. 4.

4 Simulation Results

The feasibility of a fault attack or at least its efficiency depends on the capabilities of the attacker and the type of faults he can induce. In our considerations, we use a fault model where either transient or permanent faults are induced randomly into the device. Faults are modeled as an n-bit error vector $E = \{e_n, \ldots, e_i, \ldots, e_1, e_0\}$, where $e_i \in \{0, 1\}$ and $e_i = 1$ indicate that bit i is faulty. The number of ones in this vector is equal to the number of inserted faults. Simulations are performed for bit flip faults.

In order to measure the detection capability of the proposed architectures (Figs. 4 and 5), we used VHDL hardware description language and the VHDL simulator provided by Aldec, Active HDL. The VHDL model of the block cipher has been modified with the faults. Faults were injected in the first round in different places of the cipher. Output signals have been compared to correct signals. In this way, the obtained fault coverage gives a measure of the error detection capability. In this experiment we focused on transient and permanent, single and multiple bit flips faults. Probability of error detection is shown in Table 1 for permanent faults and in

Table 1 Probability of permanent error detection—faults in encryption path

Fault type	Bit flip fault in S-boxes				
Number of errors	1	2	3	4	5
Detection percentage. Parity bits	100 %	86.5 %	100 %	97.1 %	100 %
Detection percentage. Round level CED	100 %	100 %	100 %	100 %	100 %

Table 2 Probability of transient error detection—faults in encryption path

Fault type	Bit flip fault in S-boxes				
Number of errors	1	2	3	4	5
Detection percentage. Parity bits	100 %	82.7 %	100 %	94.3 %	100 %
Detection percentage. Round level CED	100 %	100 %	100 %	100 %	100 %

Table 3 Round level CED—probability of permanent error detection

Place of errors	Encryption and decryption path				
Number of errors	2	3	6	10	16
Detection percentage	99.8 %	99.9 %	100 %	100 %	100 %

Table 4 Probability of permanent error detection after one round

Fault type	Bit flip fault in the input of cipher				
Number of errors	1	2	3	4	5
Detection percentage. Parity bits	100 %	69.6 %	100 %	78.1 %	100 %
Detection percentage. Round level CED	100 %	100 %	100 %	100 %	100 %

Table 2 for transient faults. The faults were injected into S-boxes in the encryption path.

Multiple faults were also inserted in encryption path and decryption path. In this case detection probability was less, but only for a small number of errors. Round level CED detects five or more errors with probability 100 % as it shown in Table 3.

Faults injected to the cipher input propagate quickly to the output of the cipher. Even one fault introduced to the input of the cipher in the first round of PP-2 generates after four rounds 30 and more faults at the output of the cipher. The propagation of errors in subsequent rounds is shown in Fig. 6.

Probability of detection, (after one round) of errors introduced to the input of the cipher is shown in Table 4. All faults are detected after three rounds.

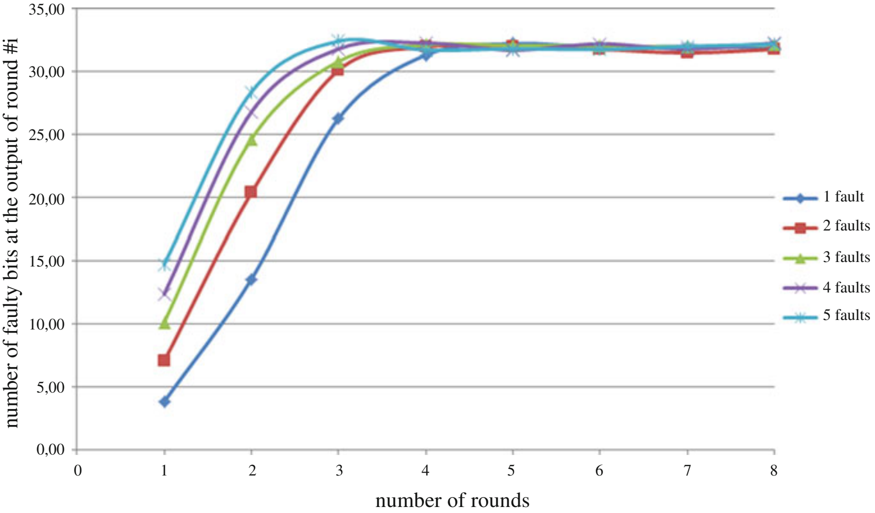


Fig. 6 Propagation of errors introduced in the first round of PP-2

5 Conclusions

Fault injection attacks on cryptographic chips are based on the observation that faults deliberately introduced into a cryptographic device leak information about the implemented algorithms. The requirements for error detection have particularly strong constraints. The number of faults necessary to perform a successful attack has been dramatically reduced during the last years. In particular, it has been shown in [6] that the AES Rijndael can be corrupted with only two faulty ciphertexts. Therefore, we proposed two schemes of error detection with moderate area overhead and with high probability of error detection.

One of them is based on parity checks for all operations and the other on the inverse relationship between encryption and decryption at the round level. Simulation experiments conducted on a large number of test cases show that these methods provide high coverage of single and multiple permanent errors which are the most common in fault attacks. These schemes offer a trade-off between area and time overhead without severely degrading the performance or modifying the encryption algorithm. They can be useful for concurrent verification of cryptographic chips especially designed for platforms with limited resources.

Acknowledgments This research was supported by 04/45/DS PB/0122.

References

1. Boneh, D., Demillo, R., Lipton, R.: On the importance of checking cryptographic protocols for faults. In: Proceedings of Eurocrypt, vol. 1233, pp. 37–51. Springer (1997)
2. Bar-El, H., Choukri, H., Naccache, D., Tunstall, M., Whelan, C.: The Sorcerer's apprentice guide to fault attacks. Proc. IEEE **94**, 370–382 (2006)
3. Idzikowska, E., Bucholc, K.: Error detection schemes for CED in block ciphers. In: Proceedings of the 5th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing EUC, Shanghai, pp. 22–27 (2008)
4. Idzikowska, E.: CED for involutional functions of PP-1 cipher. In: Proceedings of the 5th International Conference on Future Information Technology, Busan (2010)
5. Bucholc, K., Chmiel, K., Stokłosa, J.: Koncepcja szyfru blokowego PP-2. Report No. 606 (2010)
6. Piret, G., Quisquater, J.-J.: A differential fault attack technique against SPN structures, with applications to the AES and Khazad. In: Proceedings of CHES 2003. Lecture Notes in Computer Science, vol. 2779, pp. 77–88 (2003)

Secure Pseudo-Random Linear Binary Sequences Generators Based on Arithmetic Polynoms

Oleg Finko and Sergey Dichenko

Abstract We present a new approach to construction of pseudo-random binary sequences (PRBS) generators for the purpose of cryptographic data protection, secured from the perpetrator's attacks, caused by generation of masses of hardware errors and faults. The new method is based on the use of linear polynomial arithmetic for the realization of systems of boolean characteristic functions of pseudo-random sequences (PRS) generators. "Arithmetization" of systems of logic formulas has allowed to apply mathematical apparatus of residue systems for multisequencing of the process of PRS generation and organizing control of computing errors, caused by hardware faults. This has guaranteed high security of PRS generator's functioning and, consequently, security of tools for cryptographic data protection based on those PRSs.

Keywords Cryptographic data protection · Pseudo-random binary sequences · Residue number systems

1 Introduction

Pseudo-random linear sequences generators play an important role in building of communication with cryptographic data protection [1, 2]. From the list of known attacks on information security is important type of attacks, based on the generation of hardware errors and functioning of the nodes forming the binary PRS [3]. To ensure the required level of interference and fault tolerance of digital devices

O. Finko (✉) · S. Dichenko
Institute of Computer Systems and Information Security of Kuban State
Technological University, Moskovskaya St. 2, Krasnodar, Russia
e-mail: ofinko@yandex.ru; ofinko@member.ams.org

S. Dichenko
e-mail: dichenko.sa@yandex.ru

© Springer International Publishing Switzerland 2015
A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_24

developed many methods, the most common of which are backup methods and methods of error-correcting coding [4]. However, allocation methods do not provide the required levels of fault tolerance for restrictions on hardware costs, and methods of error-correcting coding is not adapted to the specifics of construction and operation means of data protection (MDP), in particular, the generators of the PRS.

2 Analysis of Attacks Based on Hardware Faults Generation

Currently, the following types of attacks on sites of formation of binary PRS are considered (attack on) [5]:

- Analysis of results of power consumption measurements;
- Analysis of results of operations performance duration;
- Analysis of accidental hardware faults;
- Analysis of intentionally generated hardware faults, etc.

The last two types of faults are not investigated enough currently and thus are threatening to the information security of the functioning of modern and perspective MDP. The origin of those attacks lies in the use of thermal, high frequency, ionizing, and other types of external influences onto MDP for the purpose of creation of masses of faults in hardware functioning by initializing of computing errors.

Hardware attacks can be divided into two classes:

1. **Direct hardware attacks** The consequences of those attacks are failures of data protection tools. There is a method of analysis of the consequences of those failures. These types of attacks mean that in distortion in the certain places of algorithm of transformation, which results in computing errors. Those errors can lead, for example, to repeated generation of the elements of PRS or in generation of faulty elements of PRS, which is unacceptable.
2. **Attacks on postfailure recovery means** Some systems do not recovery means. If the system protection is destroyed, it is impossible to restore the operational mode. That is why such systems need to have means of protection against attacks of the malefactor and to support the possibility of updating the security system without stopping the program running.

Attacks, based on errors generation by means of external influence are highly efficient for the majority of currently known and used algorithms of PRS generation. It is known that probability of error generation is proportional to the time corresponding registers has been affected by the radiation, if the registers are in favorable condition for error occurrence, and to the quantity of bits, in which the error occurrence is expected. The most widely used and proven means of creating PRS are algorithms and structures—Linear feedback shift register (LFSR)—of PRS generation, based on the use of feedback functions of logic [1, 2].

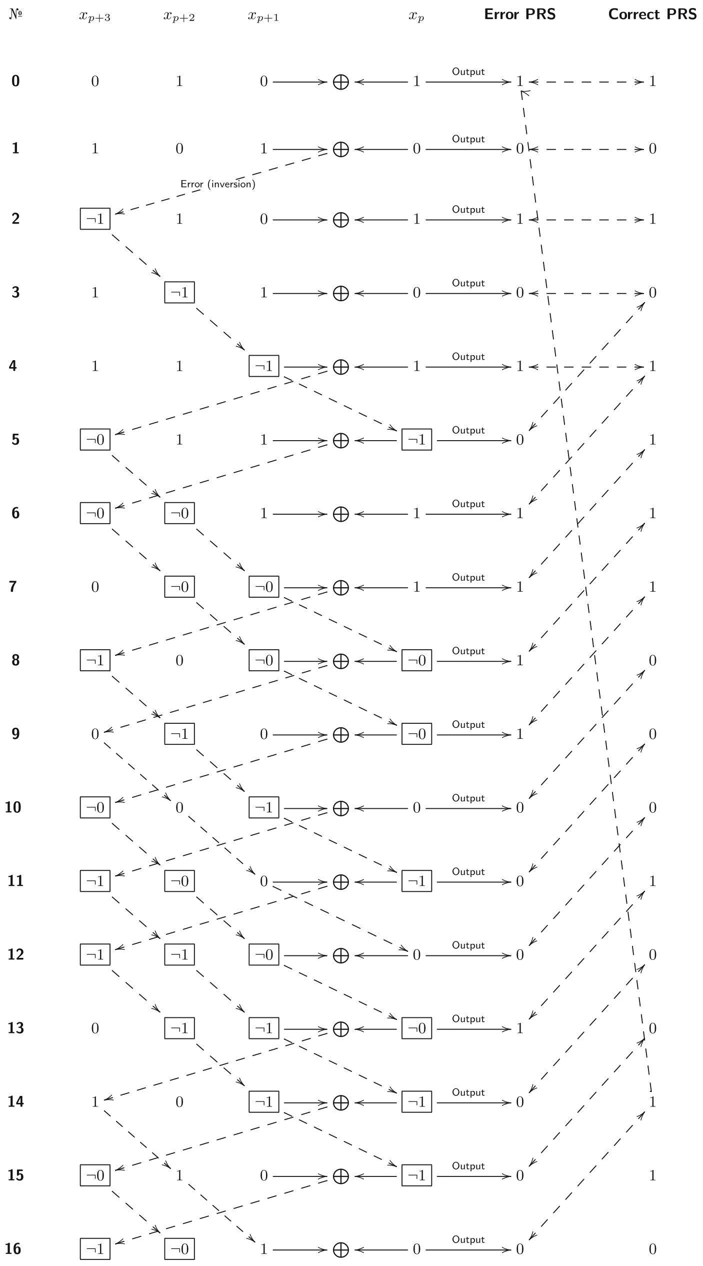

№ x_{p+3} x_{p+2} x_{p+1} x_p Error PRS Correct PRS
0 0 1 0 ⊕ 1 Output 1 1
1 1 0 1 ⊕ 0 Output 0 0
Error (inversion)
2 ¬1 1 0 ⊕ 1 Output 1 1
3 1 ¬1 1 ⊕ 0 Output 0 0
4 1 1 ¬1 ⊕ 1 Output 1 1
5 ¬0 1 1 ⊕ ¬1 Output 0 1
6 ¬0 ¬0 1 ⊕ 1 Output 1 1
7 0 ¬0 ¬0 ⊕ 1 Output 1 1
8 ¬1 0 ¬0 ⊕ ¬0 Output 1 0
9 0 ¬1 0 ⊕ ¬0 Output 1 0
10 ¬0 0 ¬1 ⊕ 0 Output 0 0
11 ¬1 ¬0 0 ⊕ ¬1 Output 0 1
12 ¬1 ¬1 ¬0 ⊕ 0 Output 0 0
13 0 ¬1 ¬1 ⊕ ¬0 Output 1 0
14 1 0 ¬1 ⊕ ¬1 Output 0 1
15 ¬0 1 0 ⊕ ¬1 Output 0 1
16 ¬1 ¬0 1 ⊕ 0 Output 0 0

Fig. 1 Example of operation of the LFSR when an error occurs (¬*x*—logical inversion *x*)

The structure of LFSR is determined by the forming polynomial:

$$D(\chi) = \chi^{\tau} + \chi^{t_l} + \cdots + \chi^{t_2} + \chi^{t_1} + 1,$$

where $\tau, t_i \in N$ and characteristic equation based on it:

$$\begin{aligned} x_{p+\tau} &= x_p \oplus x_{p+t_1} \oplus x_{p+t_2} \oplus \cdots \oplus x_{p+t_l} \\ &= c_0 x_p \oplus c_1 x_{p+1} \oplus \cdots \oplus c_{\tau-2} x_{p+\tau-2} \oplus c_{\tau-1} x_{p+\tau-1}, \end{aligned} \tag{1}$$

where $x_p, c_i \in \{0, 1\}$; $p \in N$; $i = 0, 1, \ldots, \tau - 1$; $c_{i \in \{0, t_1, t_2, \ldots, t_l\}} = 1$.

In linear algebra the next element of PRS $x_{p+\tau}$ is calculated as the following multiplication:

$$\left\| \begin{matrix} x_{p+1} \\ x_{p+2} \\ \cdots\cdots \\ x_{p+\tau-1} \\ x_{p+\tau} \end{matrix} \right\|^{\top} = \left\| \begin{matrix} 0 & 1 & \ldots\ldots & 0 & 0 \\ & & \ldots\ldots\ldots & & \\ 0 & 0 & \ldots\ldots & 1 & 0 \\ 0 & 0 & \ldots\ldots & 0 & 1 \\ c_0 & c_1 & \ldots\ldots & c_{\tau-2} & c_{\tau-1} \end{matrix} \right\|^{\top} \cdot \left\| \begin{matrix} x_p \\ x_{p+1} \\ \cdots\cdots \\ x_{p+\tau-2} \\ x_{p+\tau-1} \end{matrix} \right\|^{\top}.$$

When the described attack is performed the conditions arise for PRS modification or its repeated generation. The effect of repeated generation of a site of PRS is explained by means of Fig. 1 (the forming polynomial: $D(\chi) = \chi^4 + \chi + 1$; the characteristic equation: $x_{p+4} = x_{p+1} \oplus x_p$; the initial conditions: $x_p = 1, x_{p+1} = 0$, $x_{p+2} = 1, x_{p+3} = 0$).

Thus, those attacks, which are based on creating the conditions under which mass hardware errors occur, are threatening for MDP. One of the ways of solving this problem is development of methods for increasing the reliability of the functioning of sites of data protection tools, mostly subjected to attacks of the described type, in particular the sites of forming of the encryption algorithm (cipher), based on PRS generation.

3 Analysis of Methods for Reliable Binary PRS Generation

Currently, the required level of functional reliability of the sites of binary PRS generation is reached both by using excessive devices (reservation) and timely access by various repetitions of the calculations. In digital schemotechnics there are solutions known based on the use of methods of error-correction coding [4]. In order to use those methods for PRS generators it is necessary preliminary to solve the issue multisequencing the process of PRS calculations. The solution is based on the use of classic parallel algorithms of recursion [6].

For example, for the characteristic equation:

$$x_{p+\tau} = x_{p+t} \oplus x_p, \tag{2}$$

corresponding to treen $D(\chi) = \chi^\tau + \chi^t + 1$, it is possible to build a system of characteristic equations:

$$\begin{cases} x_{q,\tau-1} = x_{q-1,\tau-1} \oplus x_{q-1,\tau+t-1}, \\ x_{q,\tau-2} = x_{q-1,\tau-2} \oplus x_{q-1,\tau+t-2}, \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots \\ x_{q,1} = x_{q-1,1} \oplus x_{q-1,t+1}, \\ x_{q,0} = x_{q-1,0} \oplus x_{q-1,t}. \end{cases}$$

Similarly, for the general Eq. (1):

$$\begin{cases} x_{q,\tau-1} = c_0^{(\tau-1)} x_{q-1,0} \oplus c_1^{(\tau-1)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(\tau-1)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(\tau-1)} x_{q-1,\tau-1}, \\ x_{q,\tau-2} = c_0^{(\tau-2)} x_{q-1,\,0} \oplus c_1^{(\tau-2)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(\tau-2)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(\tau-2)} x_{q-1,\tau-1}, \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ x_{q,1} = c_0^{(1)} x_{q-1,0} \oplus c_1^{(1)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(1)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(1)} x_{q-1,\tau-1}, \\ x_{q,0} = c_0^{(0)} x_{q-1,0} \oplus c_1^{(0)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(0)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(0)} x_{q-1,\tau-1}, \end{cases} \tag{3}$$

where $c_i^{(j)} \in \{0, 1\}$ $(i, j = 0, 1, \dots, \tau-1)$. The principle of parallel lasing elements PRS based on (3) is illustrated by a graph (see Fig. 2).

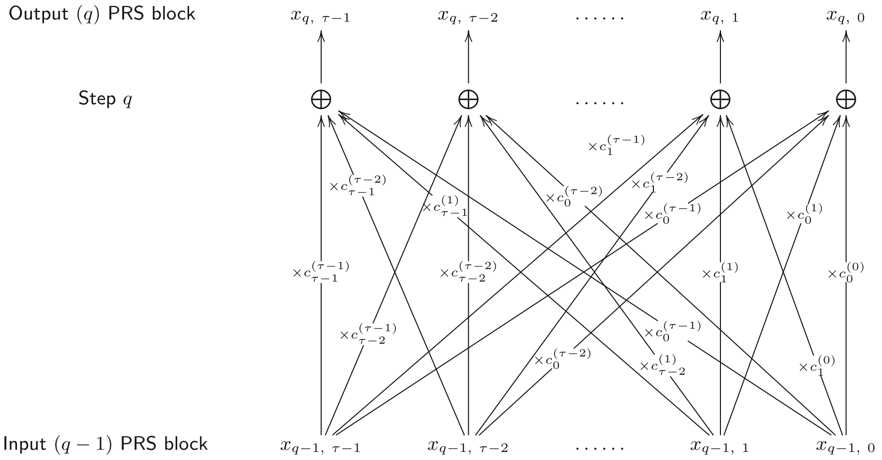


Fig. 2 Graph generating elements parallel PRS based on (3)

System (3) forms an information matrix:

$$\mathbf{G_{Inf}} = \left\| \begin{array}{ccccc} c_0^{(\tau-1)} & c_1^{(\tau-1)} & \cdots\cdots & c_{\tau-2}^{(\tau-1)} & c_{\tau-1}^{(\tau-1)} \\ c_0^{(\tau-2)} & c_1^{(\tau-2)} & \cdots\cdots & c_{\tau-2}^{(\tau-2)} & c_{\tau-1}^{(\tau-2)} \\ \multicolumn{5}{c}{\cdots\cdots\cdots\cdots} \\ c_0^{(1)} & c_1^{(1)} & \cdots\cdots & c_{\tau-2}^{(1)} & c_{\tau-1}^{(1)} \\ c_0^{(0)} & c_1^{(0)} & \cdots\cdots & c_{\tau-2}^{(0)} & c_{\tau-1}^{(0)} \end{array} \right\|^{\top}.$$

Thus we obtain the qth block of the PRS:

$$\mathbf{X}_q = \mathbf{G_{Inf}} \cdot \mathbf{X}_{q-1},$$

where

$$\mathbf{X}_q = \left[x_{q,\tau-1}\ x_{q,\tau-2}\ \ldots\ x_{q,1}\ x_{q,0}\right]^{\top},$$
$$\mathbf{X}_{q-1} = \left[x_{q-1,\tau-1}\ x_{q-1,\tau-2}\ \ldots\ x_{q-1,1}\ x_{q-1,0}\right]^{\top}.$$

Adding to the system (3) checking the equations: $\mathbf{G_{Gen}}$, consisting of the information and the check matrix by adding (3) validation expressions:

$$\begin{cases} x_{q,\tau-1} = c_0^{(\tau-1)} x_{q-1,0} \oplus c_1^{(\tau-1)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(\tau-1)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(\tau-1)} x_{q-1,\tau-1}, \\ x_{q,\tau-2} = c_0^{(\tau-2)} x_{q-1,\ 0} \oplus c_1^{(\tau-2)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(\tau-2)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(\tau-2)} x_{q-1,\tau-1}, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ x_{q,1} = c_0^{(1)} x_{q-1,0} \oplus c_1^{(1)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(1)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(1)} x_{q-1,\tau-1}, \\ x_{q,0} = c_0^{(0)} x_{q-1,0} \oplus c_1^{(0)} x_{q-1,1} \oplus \cdots \oplus c_{\tau-2}^{(0)} x_{q-1,\tau-2} \oplus c_{\tau-1}^{(0)} x_{q-1,\tau-1}, \\ x_{q,r-1}^{*} = a_0^{(r-1)} x_{q-1,0} \oplus a_1^{(r-1)} x_{q-1,1} \oplus \cdots \oplus a_{\tau-2}^{(r-1)} x_{q-1,\tau-2} \oplus a_{\tau-1}^{(r-1)} x_{q-1,r-1}, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ x_{q,0}^{*} = a_0^{(0)} x_{q-1,0} \oplus a_1^{(0)} x_{q-1,1} \oplus \cdots \oplus a_{\tau-2}^{(0)} x_{q-1,\tau-2} \oplus a_{\tau-1}^{(0)} x_{q-1,\tau-1}, \end{cases}$$

where r—the number of redundant symbols used linear code, $a_i^{(j)} \in \{0, 1\}$ $(i = 0, 1, \ldots, \tau-1;\quad j = 0, \ldots, r-1)$.

A generator matrix takes the form:

$$\mathbf{G_{Gen}} = \left\| \begin{array}{ccccc} c_0^{(\tau-1)} & c_1^{(\tau-1)} & \cdots\cdots & c_{\tau-2}^{(\tau-1)} & c_{\tau-1}^{(\tau-1)} \\ c_0^{(\tau-2)} & c_1^{(\tau-2)} & \cdots\cdots & c_{\tau-2}^{(\tau-2)} & c_{\tau-1}^{(\tau-2)} \\ \multicolumn{5}{c}{\cdots\cdots\cdots\cdots} \\ c_0^{(1)} & c_1^{(1)} & \cdots\cdots & c_{\tau-2}^{(1)} & c_{\tau-1}^{(1)} \\ c_0^{(0)} & c_1^{(0)} & \cdots\cdots & c_{\tau-2}^{(0)} & c_{\tau-1}^{(0)} \\ a_0^{(r-1)} & a_1^{(r-1)} & \cdots\cdots & a_{\tau-2}^{(r-1)} & a_{\tau-1}^{(r-1)} \\ \multicolumn{5}{c}{\cdots\cdots\cdots\cdots} \\ a_0^{(0)} & a_1^{(0)} & \cdots\cdots & a_{\tau-2}^{(0)} & a_{\tau-1}^{(0)} \end{array} \right\|^{\top}.$$

Then the qth block of the PRS with the control numbers (linear block code):

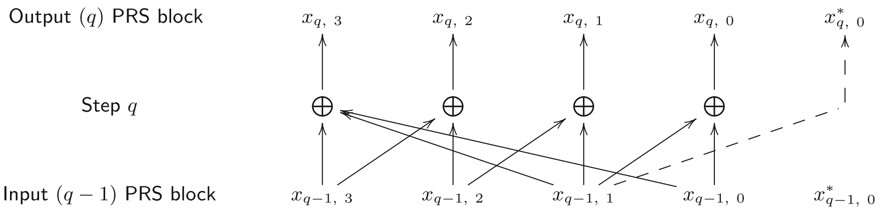


Fig. 3 Example graph parallel generation elements PRS (the characteristic equation: $x_{p+4} = x_{p+1} \oplus x_p$) error control computations (parity control)

$$\mathbf{X}^*_q = \left[x_{q,\tau-1}\ x_{q,\tau-2} \cdots x_{q,1}\ x_{q,0}\ x^*_{q,r-1} \cdots x^*_{q,0}\right]^\top$$

is calculated by:

$$\mathbf{X}^*_q = \mathbf{G}_{\mathbf{Gen}} \cdot \mathbf{X}_{q-1}.$$

Procedure error-correcting decoding is performed using the known rules [4]. The application of linear redundant codes and methods "hot" standby is not the only option for the implementation of functional diagnostics and fault tolerance of digital devices. Example graph parallel generation elements PRS error control computations is shown in Fig. 3.

Important advantages for these purposes have redundant arithmetic codes, in particular, so-called AN-codes and residue number systems (RNS) codes. The application of these codes to monitor logical data types and fault tolerance implementing devices became possible with the introduction of logical operations arithmetic expressions [7], in particular linear numerical polynomials (LNP) and modular forms [8].

4 Error Control Operation of the PRS Generators, Based on "Arithmetization" Logical Account

At the end of the last century there was formed a new direction parallel logic computation by the arithmetic (numeric) polynomials [7]. In particular received position "Modular arithmetic parallel logic computation" of the unification of the theoretical foundations of RNS [9–11] and theoretical foundations of parallel logic computation by the arithmetic of polynomials. The objective of the association is to use advantages of RNS, i.e., parallelization arithmetic, error control calculations [12] in real time and ensure high availability of computing equipment in the field of parallel logical account. In the following, these provisions were developed in various aspects, in particular, toward the implementation of cryptographic functions [13, 14]. In particular, it was considered parallel generators PRS based, in general, nonlinear (canonical) arithmetic polynomials. Use of LNP proposed by Prof. V.D. Malyugin [7] for the construction of parallel generators PRS possible to reduce the maximum length of realizing polynomial to a value of $n+1$, where n—number of arguments of a Boolean function implemented [14]. In this paper, this method is used as the basis for the construction of safe (self-checking, fault-tolerant) generators on the basis of the excess bandwidth RNS.

It is known [15] that the qth block of land PRS can be represented by a single LNP. The system of characteristic Eq. (3) must submit, as a system of Boolean functions, which in turn must be converted into a system:

$$\begin{cases} L_{\tau-1}(\mathbf{X}_{q-1}) = g_1^{(\tau-1)} x_{q-1,0} + g_2^{(\tau-1)} x_{q-1,1} + \cdots + g_\tau^{(\tau-1)} x_{q-1,\tau-1}, \\ L_{\tau-2}(\mathbf{X}_{q-1}) = g_1^{(\tau-2)} x_{q-1,0} + g_2^{(\tau-2)} x_{q-1,1} + \cdots + g_\tau^{(\tau-2)} x_{q-1,\tau-1}, \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ L_0(\mathbf{X}_{q-1}) = g_1^{(0)} x_{q-1,0} + g_2^{(0)} x_{q-1,1} + \cdots + g_\tau^{(0)} x_{q-1,\tau-1}, \end{cases}$$

where $g_j^{(i)}$ (here and then) takes the value “0” or “1” depending on the entry in the ith LNP $x_{q-1,j}$; $i, j = 0, 1, \ldots, \tau-1$.

The result of the calculation of i-LNP system appears to be a binary word of length $l_i = \lfloor \log(\sum\limits_{j=\tau-1}^{0} g_j^{(i)}) \rfloor + 1$, where $\lfloor a \rfloor$—the largest integer. Calculated total LNP:

$$\begin{aligned} L(\mathbf{X}_{q-1}) &= L_{\tau-1}(\mathbf{X}_{q-1}) + 2^{\gamma_1} L_{\tau-2}(\mathbf{X}_{q-1}) + \cdots + 2^{\gamma_{\tau-1}} L_0(\mathbf{X}_{q-1}) \\ &= g_1^{(\tau-1)} x_{q-1,0} + g_2^{(\tau-1)} x_{q-1,1} + \cdots + g_\tau^{(\tau-1)} x_{q-1,\tau-1} \\ &\quad + 2^{\gamma_1}(g_1^{(\tau-2)} x_{q-1,0} + g_2^{(\tau-2)} x_{q-1,1} + \cdots + g_\tau^{(\tau-2)} x_{q-1,\tau-1}) \\ &\quad + \cdots + 2^{\gamma_{\tau-1}}(g_1^{(0)} x_{q-1,0} + g_2^{(0)} x_{q-1,1} + \cdots + g_\tau^{(0)} x_{q-1,\tau-1}) \\ &= h_1 x_{q-1,0} + h_2 x_{q-1,1} + \cdots + h_\tau x_{q-1,\tau-1}, \end{aligned}$$

where $\gamma_k = \sum_{i=0}^{k-1}(l_i + 1)$, $k = 1, 2, \ldots, \tau-1$; $h_j \in Z$, or

$$L(\mathbf{X}_{q-1}) = \sum_{i=1}^{\tau} h_i x_{q-1,i-1}. \tag{4}$$

The final result is formed by implementing operator masking $\Xi^{\varphi}\{U\}$, which is used to determine the values of the φth Boolean function representation $U = (b_\upsilon \ldots b_\varphi \ldots b_2 b_1)_2$ (record $(\ldots)_2$ means representing a nonnegative U in a binary number), that is, $\Xi^{\varphi}\{U\} = b_\varphi$.

In RNS a nonnegative coefficient LNP (4) h_j is uniquely represented by a set of residues on the grounds RNS ($m_1, m_2, \ldots, m_n < m_{n+1} < \cdots < m_k$—pairwise simple):

$$h_j = (\alpha_1, \alpha_2, \ldots, \alpha_n, \alpha_{n+1}, \ldots, \alpha_k)_{\text{MA}}, \tag{5}$$

where $\alpha_t = |h_j|_{m_t}; t = 1, 2, \ldots, n, \ldots, k$; $|\,\centerdot\,|_m$—the smallest nonnegative deduction number $\centerdot$ on the modulo m. Operating range $M_n = m_1 m_2 \ldots m_n$ must meet $M_n > 2^s$, where $s = \sum_{1 \le \varepsilon \le \tau} l_\varepsilon$—the number of binary bits required to represent the result of a calculation LNP (4).

The remains $\alpha_1, \alpha_2, \ldots, \alpha_n$ are informational, and $\alpha_{n+1}, \ldots, \alpha_k$—are control. RNS in this case is called the extended and covers the complete set of states represented all k residues. This area is full range RNS $[0, M_k)$, where $M_k = m_1 m_2 \ldots m_n m_{n+1} \ldots m_k$, and consists of the operating range $[0, M_n)$, defined information bases RNS, and range identified redundant bases $[M_n, M_k)$, unacceptable region for the results of a calculation. This means that operations on numbers h_j are in the range $[0, M_k)$. Therefore, if the result of the operation RNS beyond M_n, it should output error calculation.

Consider RNS specified grounds $m_1, m_2, \ldots, m_n, m_{n+1}$. Each coefficient LNP h_j can be written as (5) and get redundant code RNS represented by the LNP system:

$$\begin{cases} U^{(1)} = L^{(1)}(\mathbf{X}_{q-1}) = \alpha_1^{(1)} x_{q-1,0} + \alpha_2^{(1)} x_{q-1,1} + \cdots + \alpha_\tau^{(1)} x_{q-1,\tau-1}, \\ U^{(2)} = L^{(2)}(\mathbf{X}_{q-1}) = \alpha_1^{(2)} x_{q-1,0} + \alpha_2^{(2)} x_{q-1,1} + \cdots + \alpha_\tau^{(2)} x_{q-1,\tau-1}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ U^{(n)} = L^{(n)}(\mathbf{X}_{q-1}) = \alpha_1^{(n)} x_{q-1,0} + \alpha_2^{(n)} x_{q-1,1} + \cdots + \alpha_\tau^{(n)} x_{q-1,\tau-1}, \\ U^{(n+1)} = L^{(n+1)}(\mathbf{X}_{q-1}) = \alpha_1^{(n+1)} x_{q-1,0} + \alpha_2^{(n+1)} x_{q-1,1} + \cdots \\ + \alpha_\tau^{(n+1)} x_{q-1,\tau-1}. \end{cases} \tag{6}$$

Substituting in (6) values of RNS residue on the appropriate grounds for each coefficient (4) and the values of the variables $x_{q-1,0}, \ldots, x_{q-1,\tau-1}$, get the values of LNP system (6), where $U^{(1)}, U^{(2)}, \ldots, U^{(n)}, U^{(n+1)}$—nonnegative integer. In accordance with the Chinese remainder theorem solve the system of equations:

$$\begin{cases} U^* = |U^{(1)}|_{m_1}, \\ U^* = |U^{(2)}|_{m_2}, \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ U^* = |U^{(n)}|_{m_n}, \\ U^* = |U^{(n+1)}|_{m_{n+1}}. \end{cases} \tag{7}$$

Since $m_1, m_2, \ldots, m_n, m_{n+1}$ are pairwise prime, then the only solution of (7) gives the expression:

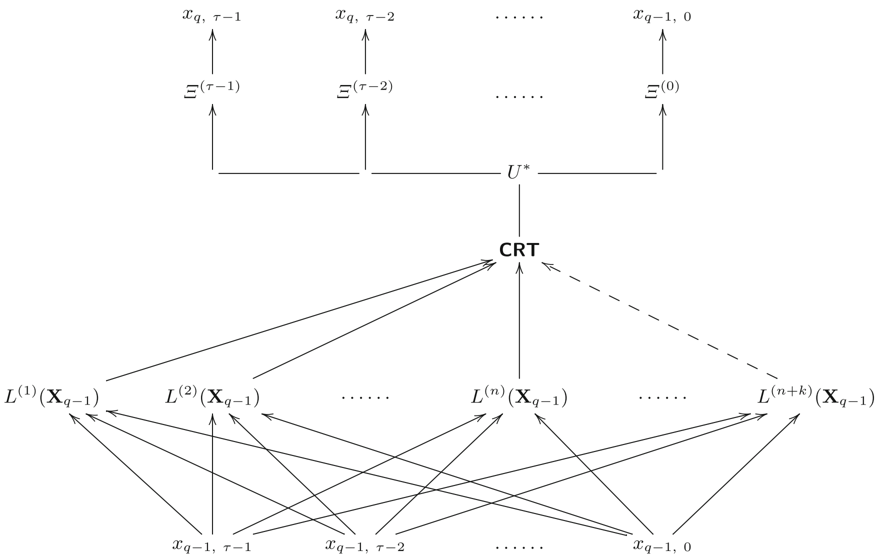


Fig. 4 Graph of parallel generation PRS based on the Chinese remainder theorem (CRT)

$$U^* = \left| \sum_{s=1}^{n+1} M_{s,n+1} \mu_{s,n+1} U^{(s)} \right|_{M_{n+1}}, \tag{8}$$

where $M_{s,n+1} = \dfrac{M_{n+1}}{m_s}$, $\mu_{s,n+1} = |M_{s,n+1}^{-1}|_{m_s}$, $M_{n+1} = \prod_{s=1}^{n+1} m_s$.

Graph parallel generation PRS based on (8) is shown in Fig. 4. The occurrence of the result of the calculation (8) in the range (control expression):

$$0 \le U^* < M_n,$$

means the absence of detectable errors of calculations.

5 Reconfiguration of Equipment

Restore reliable operation of the generator of the PRS in the case of long-term failure is possible by correcting an error or reconfiguration of equipment generator (active redundancy). The first option is unacceptable because it does not guarantee no penetration of undetectable errors in the result of the encryption. By methods of modular redundant coding is made possible to apply a variant of the reconfiguration of the equipment by excluding from the operation of the failed equipment.

Table 1 Calculation table orthogonally bases and modules RNS

j	$B_{1,j}$	$B_{2,j}$	$\dots$	$B_{n+2,j}$	M_j
1	0	$\frac{M_1\mu_{2,1}}{m_2}$	$\dots$	$\frac{M_1\mu_{n+2,1}}{m_{n+2}}$	$m_2m_3\dots m_{n+2}$
2	$\frac{M_2\mu_{1,2}}{m_1}$	0	$\dots$	$\frac{M_2\mu_{n+2,2}}{m_{n+2}}$	$m_1m_3\dots m_{n+2}$
……	………	………	$\dots$	………	……………
$n+2$	$\frac{M_{n+2}\mu_{1,n+2}}{m_1}$	$\frac{M_{n+2}\mu_{2,n+2}}{m_2}$	$\dots$	0	$m_1m_2\dots m_{n+1}$

After localization of the faulty equipment—for example—a single channel operation RNS, the reconfiguration operation is performed by the calculation U^* from the system:

$$\begin{cases} U^* = |\widetilde{U}^{(1)}|_{m_1}, \\ \dots\dots\dots\dots\dots\dots \\ U^* = |\widetilde{U}^{(n)}|_{m_n}, \\ U^* = |\widetilde{U}^{(n+1)}|_{m_{n+1}}, \\ U^* = |\widetilde{U}^{(n+2)}|_{m_{n+2}} \end{cases}$$

on the modules corresponding to the serviceable equipment of the computer:

$$U^* = |\widetilde{U}^{(1)}B_{1,j} + \widetilde{U}^{(2)}B_{2,j} + \dots + \widetilde{U}^{(n+2)}B_{n+2,j}|_{M_j},$$

where $\widetilde{U}^{(i)}$—are numbers that may contain errors; $B_{i,j}$—orthogonal bases; $i, j = 1, 2, \dots, n+2$; $i \neq j$; $B_{i,j} = \frac{M_j\mu_{i,j}}{m_i}$; $M_j = \frac{M_{n+2}}{m_j}$; $\mu_{i,j}$ is calculated from the comparison: $\frac{M_j\mu_{i,j}}{m_i} \equiv 1 \pmod{m_i}$. Compiled Table 1 contains the values of the orthogonal bases and modules of the system for the occurrence of a single error for each base RNS.

6 Conclusion

It is known that the use of RNS already with two redundant bases allows us to provide a level of fault tolerance modular transmitter that exceeds the tolerance provided by the method of rorovana equipment. These redundant hardware costs are reduced from 200 % (triple) up to 30–40 % (when using RNS) [16]. At the same time it should be noted that the amount of hardware, PRS generator operating in accordance obtained by the method, may exceed the hardware failover LFSR, built in accordance with traditional solutions. So you should make a fundamentally new level of functional flexibility of the designed generator PRS able to implement many other cryptographic functions, which are time-varying, without rebuilding the structure.

This allows for the implementation of the device not only in programmable logic integrated circuit, but also high-tech large custom integrated circuits, in particular used for the implementation of number theoretic transformations in the field of digital signal processing.

The implementation of the PRS generators using LNP and redundant RNS allows to obtain a new class of solutions aimed at the safe implementation of the logical cryptographic functions, in particular parallel generators PRS. This is provided as a functional control equipment (in real time), and its fault tolerance through reconfiguration of the structure of the evaluator in the process of its degradation. Classic LFSR considered in the present work, is the basis and more complex, for example, combining generators PRS. Use of the implementation of the PRS generator modular arithmetic provides the possibility of applying the proposed solutions in the hybrid cryptosystems (including asymmetric) [14]. When this arithmetic calculator that supports the implementation of asymmetric cryptographic algorithms may be used to implement systems of Boolean functions (elements PRS).

References

1. Forouzan, B.A.: Cryptography and Network Security. McGraw Hill (2008)
2. Schneier, B.: Applied Cryptography. Wiley, New York (1996)
3. Yang, B., Wu, K., Karri, R.: Scan based side channel attack on data encryption standard. Report **2004**(324), 114–116 (2004)
4. Hetagurov, J.A., Prudnaya, Y.P.: Improving the reliability of digital devices redundant coding methods. Energiya, Moscow (1974)
5. Kelsey, J.: Protocol interactions and the chosen protocol attack. Security protocols. In: 5th International Workshop, pp. 91–104, Springer New York. (1996)
6. Ortega, J.M.: Introduction to Parallel & Vector Solution of Linear Systems. Plenum Press, New York (1988)
7. Shmerko, V.P.: Malyugin's theorems: a new concept in logical control, VLSI design, and data structures for new technologies. Autom. Remote. Control. **65**(6), 893–912 (2004). June
8. Finko, O.A.: Large systems of boolean functions: realization by modular arithmetic methods. Autom. Remote. Control. **65**(6), 871–892 (2004). June
9. Garner, H.L.: Number systems and arithmetic. Adv. Comput. **6**, 131–194 (1965)
10. Omondi, A., Premkumar, B.: Residue Number System: Theory and Implementation. Imperial College Press, London (2007)
11. Soderstrand, M.A., Jenkins, W.K., Jullien, G.A., Tailor, F.J.: Residue Number System Arithmetic: Modern Application in Digital Signal Processing. IEEE Press, New York (1986)
12. Jenkins, W.K.: The design of error checkers for self-checking residue number arithmetic. IEEE Trans. Comput. **4**, 388–396 (1983)
13. Finko, O.A., Vishnevsky, A.K.: Parallel realization of systems of substitutions by numerical polynoms. In: Papers of the 5th International Conference Parallel Computing and Control Problems, pp. 935–943. Moscow (2010)
14. Finko, O.A., Vishnevsky, A.K.: Standard function hybrid cryptosystem arithmetic and logical multinomial realization. Theory and Techniques of Radio, pp. 32–38. Voronezh (2011)
15. Finko, O.A., Dichenko, S.A., Eliseev, N.I.: Error function generator binary PRS control implemented on arithmetic polynomials. St. Petersburg State Polytechnical University J. Comput. Sci. Telecommun. Control Syst. **176**(4), 142–149 (2013)
16. Krasnobaev, V.A.: Reliable model in the computer residue number system. Electron. Model. **7**(4), 44–46 (1985)

Part IV
Software Technologies

Energy-Efficient Network Services as Smart Grid Issue

Andriy Luntovskyy, Josef Spillner and Volodymyr Vasyutynskyy

Abstract Our high-tech twenty-first century is, in particular, also the century of “small power supply systems” due to the use of advanced information and communication technologies in energy networks. Creation of combined systems called Smart Grid opens great prospects for the development of both of these industries (energy and IT) and is intended to provide a synergistic effect through IT-brokered power generation and distribution. This overview and position paper examines existing models of Smart Grid, the suitable basic networking and service technologies, as well as typical usage scenarios for integrated intelligent networks.

Keywords Clouds · Data centers · Grid · HVAC · Kyoto protocol · Network storage · Power usage effectiveness · QoS · QoE · RAIC · RAID · Smart Grid · Smart metering · XaaS

1 Energy Efficiency of the Networks

In the course of development of networked applications and especially of hosted applications and cloud computing, the following three phases can be identified:

- *First phase for rollout of networks and Internet (about 1970–1999)* had the purpose of improving the QoS (Quality of Service). The large computing centers were economically effective due to usage of broadband Internet connections. They helped also in mitigation of DDoS (distributed denial of service) attacks due to load

A. Luntovskyy (✉)
BA Dresden University of Cooperative Education, Hans-Grundig-Str. 25,
01307 Dresden, Germany
e-mail: andriy.luntovskyy@ba-dresden.de

J. Spillner · V. Vasyutynskyy
Dresden University of Technology, Noethnitzer Str. 46, 01187 Dresden, Germany
e-mail: josef.spillner@tu-dresden.de

V. Vasyutynskyy
e-mail: volodymyr.vasyutynskyy@tu-dresden.de

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_25

distribution between several servers. The system reliability was improved due to better availability of spare parts (hard drives, power units, switches, etc.) and emergency power generators in large centers, where they were feasible.

- *In the second phase of development of Internet services (about 2000–2010)* the improvement in QoS was accompanied by cost optimization, among others, due to service virtualization (minimum costs by strictly given QoS constraints). But also the large size of computing centers still led indirectly to less costs on the side of customers due to economy of scale when buying large charges of spare parts and electricity. The maintenance costs in the large computing centers is also less than in smaller ones, because the servers are updated centrally with security patches, upgrades can be better tested before deploying, and the maintenance actions are mostly the same at all servers. Enhancing the role of clouds as integration way for Big Data and Computing Power characterizes this development phase.
- *The third phase (after 2011)* was triggered by the trend of "green" IT and increasing energy demand and prices. The computing centers were built more often in the colder regions of the Earth. For example, Google achieves the PUE (power usage effectiveness) of 1.12 due to further optimization of hardware, waste heat recycling systems, and building construction features like improved air circulation, reuse of waste heat, etc. This means that only 12 % of energy required for computing was used not by servers, but by other services like conditioning, energy distribution, lighting, surveillance systems, etc. According to Uptime Institute 2012 Data Center Survey, the average PUE in the domain was about 1.89, which means a significant improvement on the side of Google, i.e.,

$$[\mathrm{Max}(\mathrm{PUE})] \vee [\mathrm{QoS} \geq \mathrm{Constraints}] \vee [\mathrm{Costs} \leq \mathrm{Constraints}] \tag{1}$$

In the third phase where we are now (maximum PUE by strictly given QoS/cost constraints), the following options of further improving the energy efficiency will be used:

- Simultaneous operation of as few units as possible, thanks to service and resource virtualization, increased resource sharing, and load balancing.
- Better load utilization of operating units, e.g., by dynamic operation of servers, distribution of virtual machines, and scheduling.
- Using more energy-efficient units (measured in Watt per GHz) needs less energy for cooling.
- Optimized selection of location, e.g., in cold regions, close to rivers, free cooling.
- Reuse of waste heat, e.g., for building heating or warming of potable water.
- Use of a mix of local or regional energy producers to reduce transmission losses. This requires a Smart Grid and brokering, i.e., an application in the cloud, to work on a larger scale.

Structure of the work The structure of this paper is as follows. In Sect. 1, own periodization of energy efficiency of the network structures is offered. In Sect. 2, two typical scenarios for Smart Grid deployment are discussed. The services, architectures, and multilevel models are compared in Sect. 3. Section 4 finally leads into the four case studies on Smart Grid.

2 Smart Grid Deployment Scenarios

Smart Grid is a technology for the integration of electric power supply and telecommunication networks in order to increase the energy efficiency of both types of networks, reduction of CO_2 emission under the Kyoto Protocol, decentralization of existing architectures for an integrated network (i.e., one of the main principles of Internet construction), and improving its efficiency (efficient switching, and routing) under use of alternative and renewable energy sources (like wind, solar, thermal, and electromagnetic smog) combined with use of hybrid hydrocarbon-electric vehicles (PEV, Plug-in Hybrid Electric Vehicles), with optimization of network management techniques and billing services (Smart Metering) within the conventional power supply networks, as well as increasing its safety, security, and QoS in such integrated networks for power supply and telecommunication [1–5]. Active deployment of the environmentally *friendly* and thus "green" Smart Grid technology goes on today in many developed countries, for example, Australia, European Union, in particular, Germany and Austria, the USA, Canada, the People's Republic of China and South Korea, which would like to provide and reinforce its own energy independence for the future also. Several universities, for instance, Dresden University of Technology (TUD) carry out the corresponding research subjects on the mentioned area and already possess certain "know-how." The slogan of the coordinated actions for all stakeholders might be as follows: "From Internet of Data and Web Services to the Internet of Energy Services." Nowadays, there are numerous international organizations and well-known companies that are developing the technology and corresponding devices for Smart Grid. Among them are IEEE, CENELEC, Cisco, Deutsche Telekom, Siemens, etc. [6–16]. The existing basis for local area solutions of Smart Grid is built on the following well-known network technologies: Powerline, Homeplug, WiMAX, PoE (Power over Ethernet), KNX, LON (Local Operating Network), WSN (ZigBee, EnOcean), etc. [3, 4]. But there is also a necessity to develop integrative solutions for net decentralization (one of the main principles of Internet construction), to improve its efficiency, to facilitate use of alternative and renewable energy sources, and to stimulate the development of so-called efficient energy storages (batteries, peculiar energy depot) aimed to store redundant or excess (electric) energy. To reach this goal we need first to formulate a list of scientific and technical development challenges for an integrated network (Smart Grid) on the existing basis of standard network architectures, requirements for such networks, and then to develop its own basic models. How will it all work together? Let us consider the following two scenarios:

Scenario 1 What will be a middle-class network connection for an SME (small and medium enterprise) in 2020? Only one cable will provide the services such as electricity, telephone, Internet, digital high-definition television, and cloud services.

Space heating will be realized via derivation and recycling of redundant energy from multiple (virtual) servers. The wired and wireless automation local area as well as piconets like LON, KNX, ZigBee, EnOcean will be used to serve and control the indoor climate. Management of such integrated network can be performed through Ethernet LAN/WLAN links as well as convenient protocols like IP, ICMP, and SNMP. The program support, configuration, and tuning of the intelligent network are realized with use of mobile devices (smartphones and tablets), mobile apps, and through the offered Web Services/clouds [3–5, 17–23].

Scenario 2 The scenario depicts a vision inspired by the product roadmap of the German company, Siemens. According to Fig. 1, in the future Smart Grid is designed to connect four major components [6], which operate both as consumers/producers and electric energy storages. Among them are:

1. Intelligent buildings (Intelligent Home) with solar panels and local area networks for climate automation like Field Bus and WSN (Wireless Sensor Networks).
2. Enterprises for generation of (electric) energy (so-called AC Plants) based on traditional or alternative and renewable sources (like wind, solar, EM smog).
3. Electric mobility based on hydrocarbon-electric PEV that accumulates power and data and can afterward "upload" it to the network (electromobility).
4. Intelligent counters and meters (smart metering), which automates the processes, carrying out the monitoring and network management aimed to low-energy

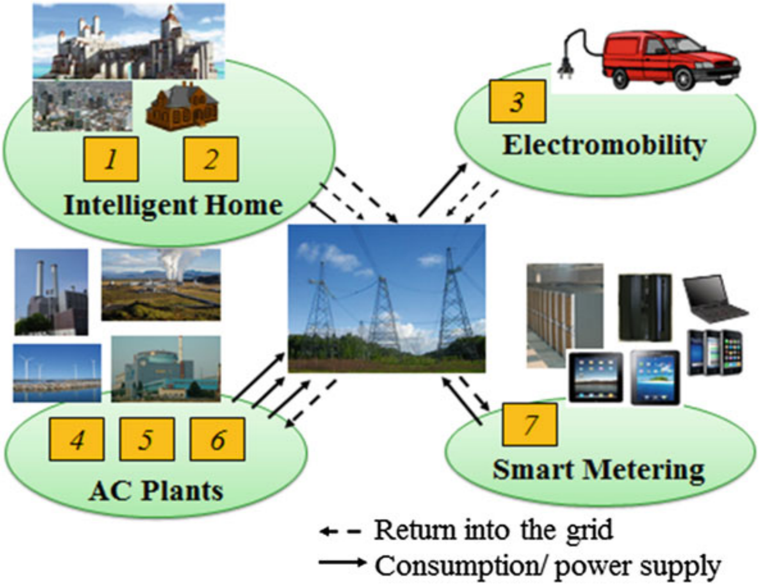


Fig. 1 Smart Grid technology highlights by [6]: *1* automated ambience; *2* HVAC; *3* PEV; *4* solar, atom, gas, hydro, and wind power plants; *5* geothermal plants; *6* combined heat and power couplings and storages; *7* on premise monitoring. Legend: *AC* Alternating current; *HVAC* Heating, ventilating, and air conditioning; *PEV* Plug-in hybrid electric vehicles

consumption on the basis of improved tariff models with respect to the workload parameters and traffic, both analog to packet-switched networks.

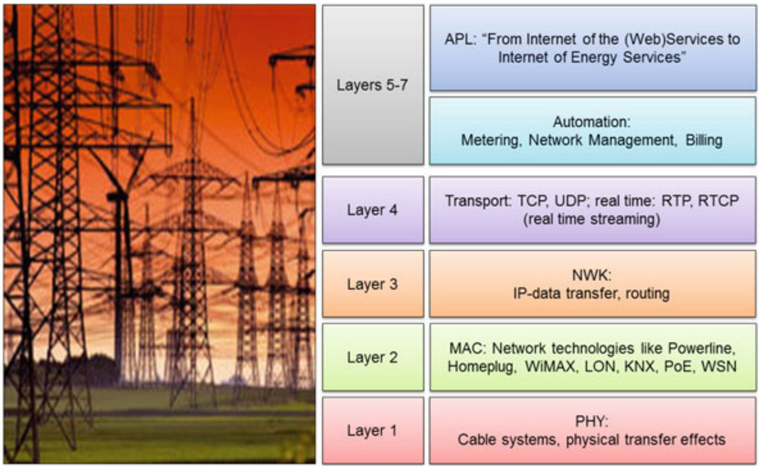
The considered components {1–4} may both use and release the excess electroenergy and stored redundant currents in the network.

3 Services, Architectures, and Multi-layered Models

The integrated architecture of Smart Grid has to repeat to a certain extent the well-known OSI network architecture (Fig. 2). But it must also be multidimensional, i.e., has to reflect not only the abstraction levels with multiple defined interfaces, functions, and services, but the various types of network technologies and domains of its use, types of consumers and service providers, device types, access control techniques, schemes for provisioning, brokering, billing, and payment for the consumed services.

Let us consider the existing multilayered and multidimensional models for Smart Grid which are oriented at shared use of telecommunications:

1. NIST Smart Grid Conceptual Model (USA).
2. IEEE Smart Grid Model.
3. A proprietary model of Cisco Smart Grid.
4. Common architecture of ITG@VDE Smart Grid (Germany).
5. Evolutionary development of model (4), the EU Smart Grid Architecture Model (European).



Layers 5-7	APL: "From Internet of the (Web)Services to Internet of Energy Services"
	Automation: Metering, Network Management, Billing
Layer 4	Transport: TCP, UDP; real time: RTP, RTCP (real time streaming)
Layer 3	NWK: IP-data transfer, routing
Layer 2	MAC: Network technologies like Powerline, Homeplug, WiMAX, LON, KNX, PoE, WSN
Layer 1	PHY: Cable systems, physical transfer effects

Fig. 2 A simplified architecture for Smart Grid. Legend: *APL* Application; *NWK* Network; *MAC* Media access control; *PHY* Physical

One of the first developed models in the area, the *Model* (*1*), is called NIST Smart Grid Conceptual Model (National Institute of Standards and Technology in the USA). It provides abstraction of properties of the integrated intelligent network based on classic three-level representation, including the following levels: 1. Power and Energy, 2. Communications, 3. IT and Computer [7].

The universal *Model* (*2*) is offered via IEEE forum. IEEE Smart Grid is a professional organization for standardization and coordination among the Smart Grid stakeholders within IEEE. Universality of the mentioned IEEE Smart Grid Model consists in creation and description of a meta-system called Smart Grid, which extends the rules, interfaces, and functions for individual intelligent networks to the so-called Smart Grid Domains, which is also based on the following three levels: 1. Power and Energy Layer, 2. Communication Layer, and 3. IT and Computer Layer. The IEEE organization shifted the focus of considering the Communication and IT and Computer Layers both (2, 3) as the determining levels for electricity distribution in Smart Grid (Power and Energy Layer) [8].

The following proprietary *Model* (*3*) was provided by the company Cisco, a network technologies company with high market share [9]. The model takes into account the development aspects of integrated (mobile) power transmission and telecommunications in the context of hardware and software that is produced via the company. Nowadays, the company Cisco provides design and implementation, deploying and supporting of infrastructure and services for Smart Grid, as well as numerous communication systems for power supply substations, automation networks (field area networks) for power supply nets, provides data security (Cisco switches, routers, firewalls ASA-CX) for the Smart Grid, creates virtual storage centers for data processing (network storages, cloud computing), thus extending those capabilities of WAN architectures. The Cisco Connected Grid Network Management Solutions (NMS) offers the infrastructure, access tools, monitoring, and management facilities for IP-able devices integrated into Smart Grid.

Furthermore, let us consider the advantages of a common architecture for Smart Grid architecture, proposed by ITG@VDE (Germany). Existing network technologies can be easily integrated into the framework of *Model* (*4*). The installed services are independent of the basic network infrastructure (refer OSI). Common architecture for Smart Grid allows adequate modeling of integrated networks of energy and information supply at different levels of abstraction. *Model* (*4*) of Smart Grid can be used recursively or hierarchically to describe the interoperability between different providers offering their services (Fig. 3):

(a) (mobile) communication;
(b) electrical energy supply;
(c) Smart Metering (intelligent control and telemetry);
(d) Smart Power Web Services.

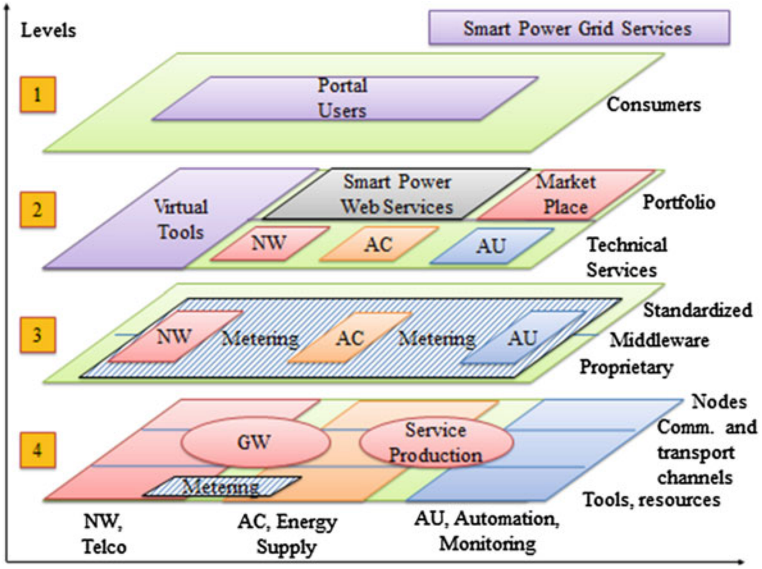


Fig. 3 Common four-layer architecture for Smart Grid [10–13] and the types of energy supply and data supply services: *1* consumers; *2* services and virtualization; *3* info-objects and service communication; *4* infrastructure/PHY. Legend: *GW* Gateway; *AC* Alternating current (energy supply nets); *AU* Automation (and management) networks; *SPGWS* Smart *Power Web Services*; *NW* Network; Metering control and telemetry; Market place allocation and reselling of services

The presence of the common architecture of Smart Grid provides nevertheless a wide field for activities and describes the ability of the model to innovations [10–13]. As the further development of this well-known and recognized *Model* (*4*), a more complex multidimensional European *Model* (*5*) called EU Smart Grid Architecture (Fig. 4) should be considered.

The model possesses its five component layers as follows: Business, Function, Information, Communication, and Component. There are two additional dimensions called Domains and Zones [14, 15]. The European Commissions on networks, communications, and technology in Brussels also believe that Smart Grid will play an important role in increasing the meaning of renewable and alternative energy sources for low-energy consumption, delivery savings, and CO_2 emission decreasing. Without integration between telecommunication and information networks the established goals are unattainable. Smart Grid is therefore a significant part of a long-term research and technology development program called Horizon 2020 [16].

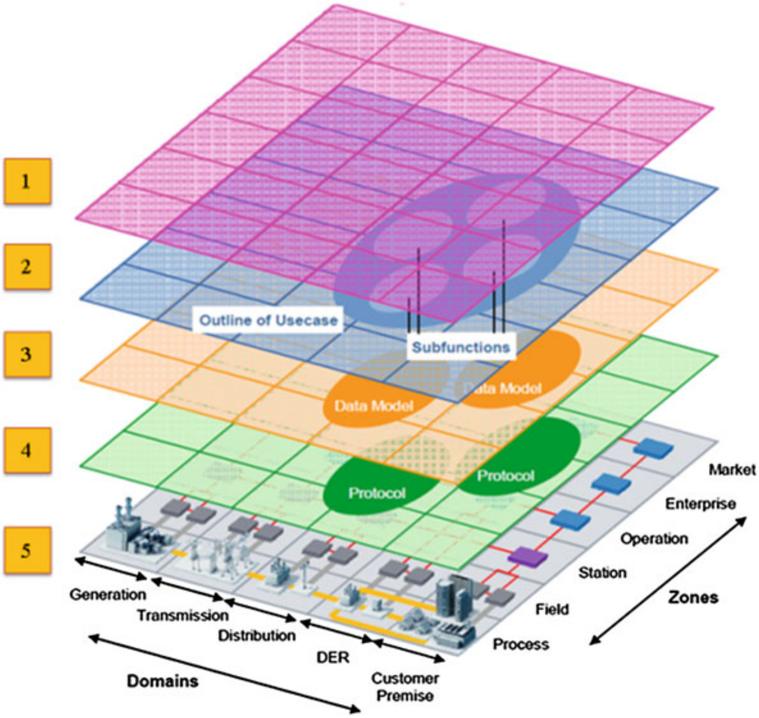


Fig. 4 EU Smart Grid Model [14, 15]: *1* Business layer; *2* Function layer; *3* Information layer; *4* Communication layer; *5* Component layer. Legend: Domains: *DER* Distributed Energy Resources; *GTD* Generation, Transmission, Distribution (production); *CP* Customer Premise (delivery); *Zones* Process, Field, Station, Operation, Enterprise, Market (PFSOEM)

4 Case Studies on Smart Grid

Case Study 1: **Smart Grid based on Powerline** Let us consider the trends in Smart Grid systems in Germany. The German Association of Electrical and Electronics Engineers VDE (in German "Technisch-wissenschaftlicher Verband der Elektrotechnik und Elektronik") insists on planned efforts for transforming the traditional electricity networks and the creation of intelligent nets. In several European countries, this approach has become a significant part of the national energy policy. In this case, it is not about some individual decisions for "several thousand kilometers of cable or 100 million euros" but integrated solutions for the Smart Grid must be developed in middle term. The main objective is as follows: reconstruction, flexibility of the entire system, redesign with elements of the modernization of infrastructure, increasing of capacity and number of power plants [10–13]. For example, from the viewpoint of R. Lehnert affiliated with Telecommunication Department at Dresden University of Technology [17], "…in a "greener" world renewable energy sources are the key to reduce the CO_2 footprint. These energy sources are typically nonstationary. This factor requires much more complex control of the grid. To enable this, the energy distribution network has to become more intelligent due to new services, distributed generation of energy (virtual power plants), and new safety and security requirements. It will finally be a Smart Grid. Nowadays, new demands on reliability and security to the support communication network appear. The discussed approach enables close system integration, optimal distributed power generation via virtual power plants, efficient control on the electricity distribution, and deployment of new network services, which are becoming more intelligent simultaneously. It has been proven that particular attention should be paid under current conditions for the deployment and use of PLC (Powerline Communications) technology (Fig. 5).

Case Study 2: **Home Automation** The Smart Grid principles like distributed functionality, integration of control, and communication have found their way quite early in the Home Automation systems. Usually, these systems provide HVAC (heating, ventilation, and air-conditioning) functionality in the rooms and buildings. The Smart Grid principles promise here a row of opportunities depicted as [3, 6, 11, 13, 17, 24, 25]:

(1) The first area includes lowering Total Cost of Ownership due to less costs of hardware. So, when using the power line connection where the power lines are used for communication between automation nodes, the additional costs for wiring are not necessary. The same is in the case of wireless sensors which do not need wiring, but depend on energy sources like batteries or collect the energy from their environment (energy harvesting).

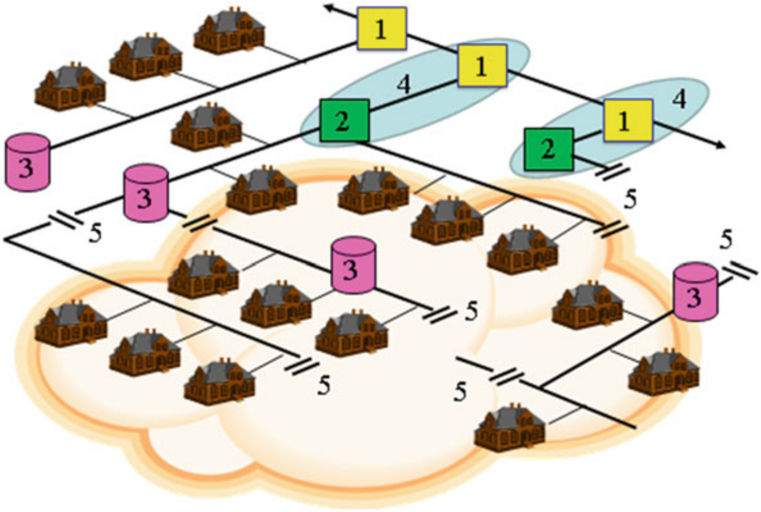


Fig. 5 Smart Grid representation as a Powerline Communication System: *1* MV part of substations; *2* LV part of substations; *3* street cabinets; *4* substations (MV + LV); *5* interruptions (open meshes). Legend: *LV* low voltage; *MV* middle voltage

(2) The second area provides new services enabled by networking of nodes and systems. For example, smart metering allows collection of live energy consumption data, by which the quicker reaction of the generators on the changing power consumption demand and avoiding of expensive peak loads is possible. Moreover, the optimization is possible also on the building level. For example, by learning the consumption profiles of home appliances, the optimization of the consumption behavior is possible. So the washing machine may start working in the night when the energy prices are lower, or the backup may run in the night when it is colder. This functionality may be combined also with other services like ambience assisted living, where the equipment consumption profiles may help in detection of emergency situations.

As regards Energy-Efficient Network Services, there are two general possibilities to optimize the energy efficiency for High-Performance Computing clusters and Big Data centers:

- Improving cooling processes;
- Workload and capacity increasing (Fig. 6).

Case Study 3: **Energy-Efficient RAIC with Mobile Access** A next constructive idea is the deployment of redundant cloud arrays (stripe and parity based dispersion). One of the possible ways for solving problems of Big Data offers intelligent combination of well-known commercial storage clouds with the use of efficient cryptographic methods and stripes/parity dispersal functionality for authenticated, transparently

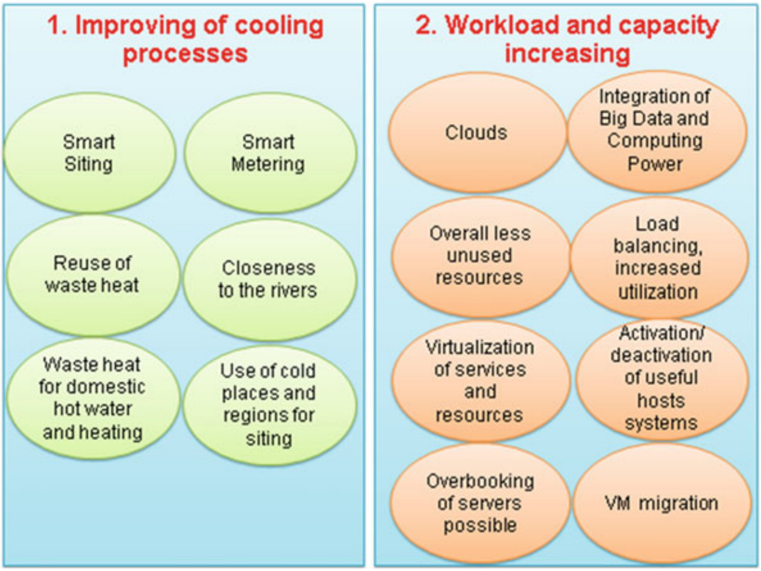

1. Improving of cooling processes
Smart Siting
Smart Metering
Reuse of waste heat
Closeness to the rivers
Waste heat for domestic hot water and heating
Use of cold places and regions for siting
2. Workload and capacity increasing
Clouds
Integration of Big Data and Computing Power
Overall less unused resources
Load balancing, increased utilization
Virtualization of services and resources
Activation/ deactivation of useful hosts systems
Overbooking of servers possible
VM migration

Fig. 6 Energy efficiency measures for high-performance computing and Big Data

encrypted, and reliable data backups. This approach is broadly known as RAIC (redundant arrays of independent clouds) [5] or simply multi storage service systems. Current RAIC concepts are optimized for minimum access time, minimum failure probability, maximum volumes, minimum costs [9, 19–23]. However, the construction can also be performed with energy efficiency as target parameter, especially in combination with access from mobile devices in a global efficiency calculation.

Energy efficiency in cloud storage controllers can be broken down into the (negligible) setup, service selection, signup, and configuration/reconfiguration processes, which typically do not happen more than once per device power-on session, and the service usage processes for storing and retrieving data. Measuring the energy efficiency of algorithms requires specialized equipment [22]. The electrical power consumption is not linear to the performance, but grows along with it, hence a performance comparison assuming equal processor load can be used for a first estimation. For a detailed power consumption analysis, we made use of the HAEC–Highly Adaptive Energy-Efficient Computing measurement infrastructure, as shown in the photo in Fig. 7. Performance characteristics of RAIC integration techniques based on [22, 23] are summarized in Table 1. Imperfect networking usability mandates intelligent

Fig. 7 HAEC laboratory measurement equipment

Table 1 Qualitative comparison of performance characteristics for versatile RAIC integration techniques

Technique	Read performance (%)	Write performance (%)
RS erasure code, 0 % redundancy, XOR	100	100
RS erasure code, 0 % redundancy, SIMD	270–1,200	270–1,200
RS erasure code, 50 % redundancy, $n = 3$	100	67
AONT-RS, $n = 3$	33	33

Legend Multicore CPU like SIMD-single instruction, multiple data by Flynn; Access coding via erasure Reed-Solomon code; Access encryption via AONT (All-or-nothing-transform encryption by Rivest) use of caching and scheduling so that slow or broken links will show no or little effect on the user of a RAIC. This typically differs per implementation. However, already on the algorithmic level, some erasure codes have been more optimized for storage, retrieval, and repair than others. Researchers have identified suitable algorithms through experiments [22]. Based on these observations, we can assume that the use of processor-specific erasure codes is beneficial for mobile devices [20, 22]. Both the device's energy efficiency and the imperfect networking usability can be tremendously improved by placing the RAIC integration onto a trusted local network proxy. So-called storage integrators can serve multiple users and enforce group policies. On the other hand, they have drawbacks concerning the trust, mobility, and overall energy efficiency given that such additional devices will remain idle for long durations. Figure 8 shows both possible integration approaches in a comparison architecture scheme [20, 22], including a mobile client integration.

Suitable software architecture for the realization of a mobile RAIC over both local and cloud storage resources is depicted via Fig. 9, following the design proposed for

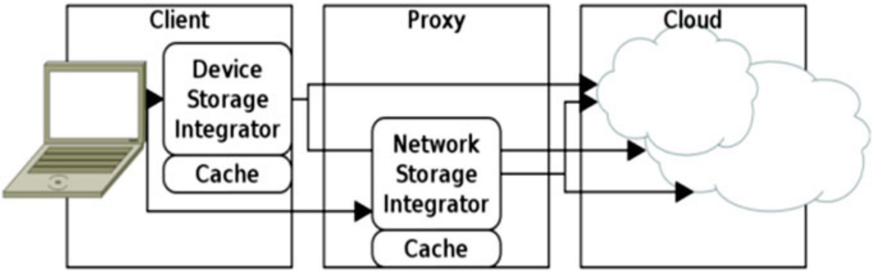


Fig. 8 Variants for efficient placement of RAIC integrator between the clouds

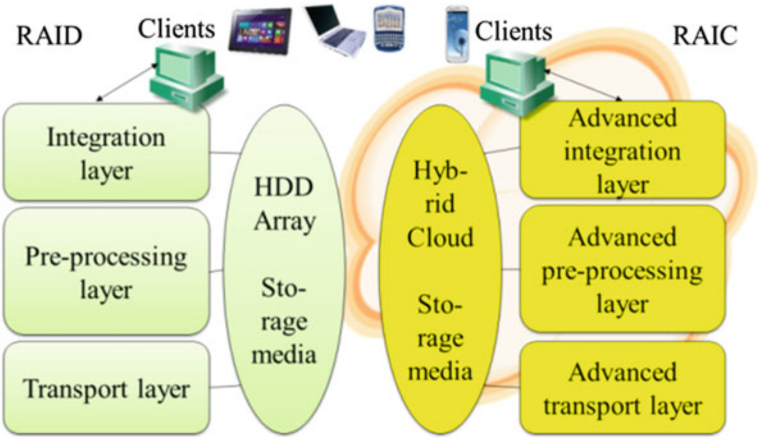


Fig. 9 Offered software architecture for realization of a RAIC: Hard disk drive (HDD), or other local drives including SD media; redundant arrays of independent clouds (RAIC)

generic cloud storage controllers [14]. The predominant client-side software for RAICs consists of the following three layers with the related functionality:

1. Integration layer: logical partition and interface to the backup application.
2. Preprocessing layer: stripes/parity dispersal routine, encryption, and other modifications.
3. Transport layer: block transfer.

The clients obtain the possibility of reliable and efficient access to an array of virtualized storage media, offered as a service or as local complementary media, with added organizational and spatial independence. This software considers the state of the art.

The offered software-layered architecture realizes a RAIC concept [10, 19–21] and includes the following already known components with the extended functionality:

1. Advanced integration layer: A local virtual file system interface available to all applications. Depending on the operating system, there may be additional specific interfaces, for instance the registration as content provider on Android or the export as RESTful Web service through RestFS.
2. Advanced preprocessing layer: Codecs: classification of document types and coding (text files, MPEG, PDF); Policies on the data storage subjects and paths; Stripes/parity dispersion routines; Authentication with MD/RSA/PKI; Encryption with AES/RSA/PKI.
3. Advanced transport layer: Parallel and block-wise streaming; Caching and local persistence; Adapters for multiple provider APIs.

Case Study 4: **Energy Recycling in Data Centers** Owing to use of today’s powerful high-end servers within the contemporary data centers with the installed broadband optical links (so-called Fiber Channel), a significant amount of heat stands out as a harmful by-product. Some companies occupy themselves already with the mentioned problem and are developing their own solutions for the disposal of heat excesses for, e.g., domestic heating and air-conditioning facilities HVAC (heating, ventilating and air-conditioning). Among them are the company AoTerra GmbH in Dresden (Germany) and data centers connected to the Helsinki utility services network. Firm AoTerra developed several corresponding products and solutions (Fig. 10), inter alia there are so-called AoCloud (own virtualized data center) and AoHeat (own smart grid) [18].

The clients use the indoor located services of virtual computing centers, the standardized XaaS cloud services like Infrastructure as a Service (IaaS), Software as a Service (SaaS), and Platform as a Service (PaaS), as well as other applications like cloud computing (Computing Service, RAID/ RAIC, SAN, NAS, Open Stack, Webhosting, Virtual OS, Own Cloud) [19, 20]. Redundant heat as a “by-product of processing” is withdrawn via server 19-racks in the energy storage, which provides circulation of hot water in the pipes within a building and heating of potable water. The central system for HVAC facilities is supported via use of PoE (Power Over Ethernet), as well as wired and wireless automation local area and piconets like

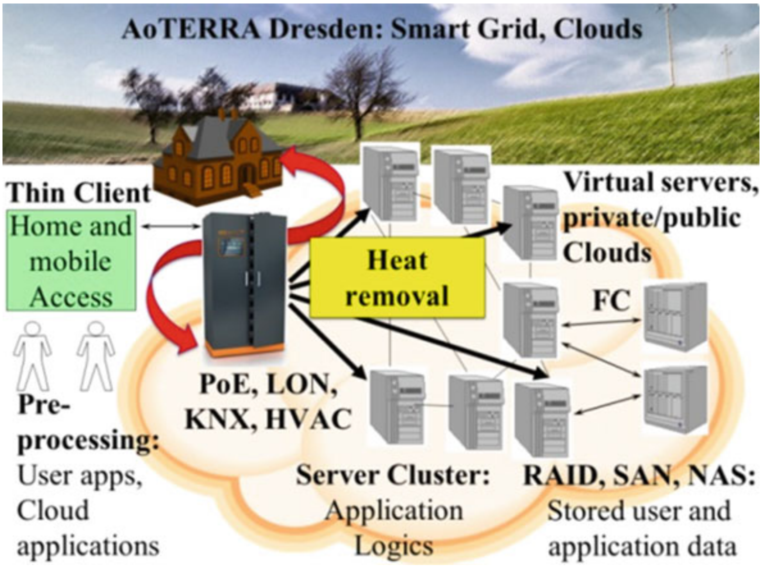

AoTERRA Dresden: Smart Grid, Clouds
Thin Client
Home and mobile Access
Pre-processing:
User apps, Cloud applications
Heat removal
PoE, LON, KNX, HVAC
Virtual servers, private/public Clouds
FC
Server Cluster:
Application Logics
RAID, SAN, NAS:
Stored user and application data

Fig. 10 Redundant heat and energy recycling in the systems of Smart Grid/Cloud Computing on the example of AoHeat [18]. Legend: *PoE* Power Over Ethernet; *FC* Fiber Channel; *HVAC* Heating, Ventilating, and Air-Conditioning; *KNX* (Konnex, former European Installation Bus); *LON* Local Operating Network

LON, KNX, ZigBee, EnOcean [1–5, 26]. The mentioned technical solution provides a higher PUE value (Power Usage Efficiency) up to 0.95 (cp. with the conventional Grid/ Cloud solutions, where it is necessary to remove the excess heat as by-product, to install more air-conditioning devices and provide them with power supply).

5 Conclusions

In some developed countries, an integrated intelligent network on the sample of conventional Internet is rapidly created (a net with Open Mesh Platforms for Energy Services). The network possesses ability to use standardized software interfaces, as well mobile apps with offered Web Services and Cloud Services. Thanks to the standardization of Smart Grid (accordingly to the intentions of the organizations like NIST, IEEE, VDE, CENELEC, etc.) software and hardware-independent access and communication between the components are guaranteed. The standardization of the structure of the open networks toward Smart Grid is today one of the development priorities as for energy and telecommunications industry in both the USA and Europe. The combined services of such networks will find in the near future (about 2020–2030) an opportunity to attract a stabile increasing number of stakeholders and users. Nowadays, there is the opportunity to create a large range of its own "smart applications" and "smart services" within the Smart Grids. Thus, to the development of such integrated electric power networks and telecommunications both will soon be given a necessary impulse. The Smart Power Grid Services (i.e., electricity) will be freely delivered, disposed to the market, and freely handled there (purchase, sale, exchange, credit, providers, resellers, etc.).

References

1. Momoh, J.: Smart Grid: Fundamentals of Design and Analysis. Wiley, New York (2012)
2. Guy, S., Marvin, S., Medd, W., Moss, T.: Urban Infrastructure in Transition: Networks, Buildings, Plans. Earthscan/Routledge, London (2012)
3. Luntovskyy, A., Guetter, D., Melnyk, I.: Planung und Optimierung von Rechnernetzen: Methoden, Modelle, Tools für Entwurf, Diagnose und Management im Lebenszyklus von drahtgebundenen und drahtlosen Rechnernetzen 411 (2011). Hand book. - Springer/Vieweg + Teubner Wiesbaden, German (ISBN: 978-3-8348-1458-6)
4. Luntovskyy, A., Klymash, M., Semenko, A.: Distributed services for telecommunication networks: Ubiquitous computing and cloud technologies 368 (2012). Monograph. - Lviv: Lvivska Politechnika, Ukrainian (ISBN: 978-966-2405-87-3)
5. Luntovskyy. A.: Distributed applications technologies 474 (2010). Monograph. - Kiev: DUIKT Publisher, Ukrainian (ISBN: 978-966-2970-51-7)
6. Siemens, A.G.: (Online 2013, in German): http://www.siemens.com/
7. NIST Framework and Roadmap for Smart Grid Interoperability Standards, Rel. 2.0: National Institute of Standards and Technology, Report 1108R2 (Online Feb. 2012, in USA): http://www.nist.gov/smartgrid/upload/NIST_Framework_Release_2-0_corr.pdf
8. IEEE Smart Grid Conceptual Model. IEEE Org., (Online 2011): http://smartgrid.ieee.org/
9. CISCO Grid Operation Solutions (Online 2013, in German): http://www.cisco.com/
10. VDE—Technisch-wissenschaftlicher Verband der Elektrotechnik und Elektronik (Online 2013, in German): http://www.vde.com/
11. Energieinformationsnetze und -Systeme: Bestandsaufnahme und Entwicklungstendenzen, ITG@VDE, p. 128, December 2010 (in German)
12. T-Systems Multimedia Solutions of Dt. Telekom (Online 2013, in German): http://www.t-systems-mms.com/
13. Benze, J.: Smart grid: normung und standardisierung. T-Systems Multimedia Solutions, Dresden, p. 48 (Online 2012, in German): http://files.hanser-tagungen.de/docs/20120724135716_Benze_CV_Abstract.pdf
14. Comité Européen de Normalisation Électrotechnique (Online 2013, in French): http://www.cencenelec.eu/
15. Smart Grid Reference Architecture/EU-CEN-CENELEC-ETSI SG Coordination Group, p. 107, Report M/490 (Online November 2012, Brussels): http://gridscientific.com/images/Smart_Grid_Reference_Artichtecture.pdf
16. EU Commission: Expert group on the security and resilience of communication networks and information systems for smart grids (Online 2013): http://www.smartgrids.eu/
17. Lehnert, R.: Smart Grid Communications. In: Proceedings of IEEE ELNANO Conference 2013, Kiev, p. 4 (April 2013)
18. AoTerra (Online 2013, in German). https://www.aoterra.de/
19. Luntovskyy, A., Guetter, D.: A concept for a modern virtual telecommunication engineering office. Intern. Res. J. Telecommun. Sci. **3**(1), 15–21 (2012). Kiev (ISSN 2219–9454)
20. Luntovskyy, A., Vasyutynskyy, V., Spillner, J.: RAICs as advanced cloud backup technology in telecommunication networks. Intern. Res. J. Telecommun. Sci. **4**(2), 30–38 (2012). Kiev (ISSN 2219–9454)

21. Semenko, A., Luntovskyy, A.: Multiservice mobile platforms of info-communications: in scientific magazine “Zviazok Communication” accredited by Superior Attestation Commission of Ukraine, VAK Ukraine, vol. 103, no. 3, pp. 7–16. Kiev (2013) (in Russian)
22. Luntovskyy, A., Spillner. J.: RAIC integration for network storages on mobile devices. In: 7th IEEE International Conference on Next Generation Mobile Apps, Services and Technologies NGMAST-2013, Prague, pp. 142–147 (September 2013) (IEEE Xplore)
23. Luntovskyy, A.: Modern Transformations in Architectures of Distributed Systems.: In International Conference SPTEL-2013 at National University “Lvivska Politechnika”, p. 8, 30 October–2 November (2013)
24. Ploennigs, J., Vasyutynskyy, V., Kabitzsch, K.: Comparative Study of Energy-Efficient Sampling Approaches for Wireless Control Networks. TII’2010, IEEE Trans. Ind. Inf., **6**(3), pp. 416–424 (August 2010) (IEEE Xplore)
25. Vasyutynskyy, V., Kabitzsch, K.: Event-based Control: Overview and Generic Model. WFCS’2010. In: IEEE International Workshop on Factory Communication Systems, Nancy, pp. 271–279, 18–21 May, (2010)
26. Luntovskyy, A., Klymash, M.: Data Security in Distributed Systems, Monograph, Lvivska Politechnika, Lviv, p. 464 (2014) (in Ukrainian)

Perfectly Nested Loop Tiling Transformations Based on the Transitive Closure of the Program Dependence Graph

Wlodzimierz Bielecki and Marek Palkowski

Abstract A novel approach to producing tiled code for perfectly nested loops is presented. It is based on the transitive closure of the program dependence graph. The approach is derived via a combination of the Polyhedral and Iteration Space Slicing frameworks that allows us to enlarge the effectiveness of the tiling transformation. The results of the evaluation of the effectiveness of a presented algorithm and the efficiency of tiled codes produced by means of the algorithm are discussed.

Keywords Optimizing compilers · Tiling · Transitive closure · Dependence graph · Code locality

1 Introduction

Tiling is a very important iteration reordering transformation for improving both data locality and extracting loop parallelism. In this paper, we consider only the first task. Tiling for improving locality groups loop statement instances in a loop iteration space into smaller blocks (tiles) allows reuse when the block fits in local memory.

For perfectly nested loops, tiling is valid when a nest of loops is fully permutable [8], i.e., when all correspondingly permuted dependence vectors have nonnegative elements.

To our best knowledge, well-known tiling techniques are based on linear or affine transformations of program loops [6–8, 11, 20]. In this paper, we present a novel approach for generating tiled code for affine loops which is based on the transitive

W. Bielecki · M. Palkowski (✉)
Faculty of Computer Science and Information Systems,
West Pomeranian University of Technology in Szczecin, Zolnierska 49,
71210 Szczecin, Poland
e-mail: mpalkowski@wi.zut.edu.pl
URL: http://www.wi.zut.edu.pl

W. Bielecki
e-mail: wbielecki@wi.zut.edu.pl
URL: http://www.wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_26

closure of the program dependence graph. We demonstrate that such an approach allows producing tiled code when techniques based on affine transformations really fail to produce any tiled code. The proposed approach allows producing tiled code even when there does not exist an affine transformation allowing the production of a fully permutable loop nest.

The contributions of this paper over previous work are as follows: (i) an algorithm demonstrating how the Iteration Space Slicing framework can be combined with the Polyhedral Model framework to enlarge the scope of the applicability of tiling transformations; (ii) clarification that this improvement is due to the fact that the presented algorithm can be directly applied to bands of loops not being fully permutable; and (iii) an evaluation of the effectiveness of the presented algorithm and the efficiency of tiled codes produced by that algorithm.

2 Background

In this paper, we deal with affine loop nests where, for given loop indices, lower and upper bounds as well as array subscripts and conditionals are affine functions of surrounding loop indices and possibly of structure parameters (defining loop indices bounds), and the loop steps are known constants.

Dependences available in the loop nest are represented with a dependence relation of the form [*input list*]→[*output list*]: *formula*, where *input list* and *output list* are the lists of variables and/or expressions used to describe input and output tuples, and *formula* describes the constraints imposed upon input and output lists and it is a Presburger formula built of constraints represented with algebraic expressions and using logical and existential operators.

In presented algorithms, standard operations on relations and sets are used, such as intersection (∩), union (∪), difference (−), domain (dom R), range (ran R), relation application ($S'= R(S)$: $e'\in S'$iff exists e s.t. $e\to e'\in R$, $e \in S$). The positive transitive closure for a given relation R, R^+, is defined as follows:

$$R^+ = \{e \to e' : \ e \to e' \in R \vee \exists e'' s.t.\ e \to e'' \in R \wedge e'' \to e' \in R^+\}. \quad (1)$$

It describes which vertices e′ in a dependence graph (represented by relation R) are connected directly or transitively with vertex e.

Transitive closure, $R*$, is defined as follows [10]:

$$R^* = R^+ \cup I, \quad (2)$$

where I is the identity relation. It describes the same connections in the dependence graph (represented by R) that R^+ does plus connections of each vertex with itself.

3 Tiling Algorithms

Our goal is to transform a loop nest of depth *d* below,

```
for(i1=lb1; i1<=ub1; i1++)
  for(i2=lb2; i2<=ub2; i2++)
    .........................
      for(id=lbd; id<=ubd; id++)
        {S}
```

to the following valid tiled loop nest

```
for(ii1=0; b1*ii1+lb1<=ub1; ii1++)
  for(ii2=0; b2*ii2+lb2<=ub2; ii2++)
    ............................
      for(iid=0; bd*iid+lbd<=ubd; iid++)
        for(i1'=.....)
          for(i2'=.....)
              ..............................
                for(id'=.....)
                  {S'}
```

where $i_1, i_2, \ldots, i_d$ are the original loop indices; $ii_1, ii_2, \ldots, ii_d$ are the loop indices defining the identifier of a tile; $i'_1, i'_2, \ldots, i'_d$ are the indices of the tiled loop nest; the constants $b_1, b_2, \ldots, b_d$ define the tile size; $lb_1, lb_2, \ldots, lb_d$ and $ub_1, ub_2, \ldots, ub_d$ state for the lower and upper bounds of the loop indices, respectively; $\{S\}$ and $\{S'\}$ denote the original and target loop nest statements, respectively.

A valid tiled loop means that all dependences of the original loop are honored in the tiled loop nest.

To illustrate how the transitive closure of the program dependence graph can be applied to produce valid tiled loops, let us consider the following working example. *Example 1*

```
for(i=0; i<=3; i++){
 for(j=0; j<=3; j++){
   a[i][j] = a[i][j+1] + a[i+1][j] + a[i+1][j-1];
  }
}
```

In this paper, we use the syntax of the Barvinok tool [16] to present results of calculations on relations and sets.

The following three relations describe all the dependences in the working loop nest.

```
R1 := {[i,j] -> [i,j+1] : 0 <= i <=3 and 0 <= j <= 2};
R2 := {[i,j] -> [i+1,j] : 0 <= i <=2 and 0 <= j <= 3};
R3 := {[i,j] -> [i+1,j-1] : 0 <= i <= 2 and 1 <= j <= 3}.
```

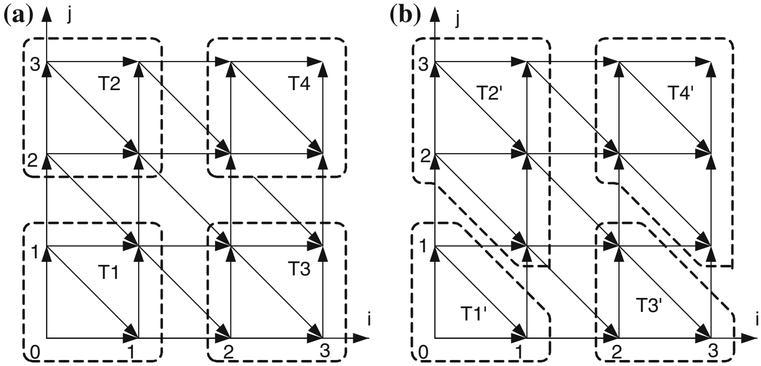


Fig. 1 **a** Original tiles for the working example. **b** Target tiles for the working example

Figure 1a shows the dependence graph for the working loop nest, where vertices represent iterations of the loop for the 4 × 4 iteration space; edges show dependences among iterations; the squares, depicted with the dashed lines, represent sets of iterations forming tiles T1, T2, T3, and T4 of size 2 × 2.

It is obvious that scanning these tiles and iterations within each tile in the lexicographic order is invalid because of the violation of the valid execution of dependent iterations (to honor a dependence, we should first execute the source of this dependence, then its destination). For example, iteration (1, 1)—the destination of the dependence (0, 2)→(1, 1)—will be executed before iteration (0, 2)—the source of this dependence. To cope with such a problem, we may change the content of the tiles as follows. We remove iteration (1, 1) from T1 and add it to T2, remove iteration (3, 1) from T3 and add it to T4. After these changes, we get the tiles T1′, T2′, T3′, T4′ presented in Fig. 1b. Now scanning these new tiles and iterations within each tile in the lexicographic order is valid.

To carry out such a modification of tiles in a formal way, we may proceed as follows. Let indices ii and jj define the identifier of a rectangular tile represented with a parametrized (with respect to indices ii, jj) set TILE. We form two additional sets, TILE_GT and TILE_LT. The first one is to contain all the iterations that are contained in tiles whose identifiers are lexicographically greater than that of set TILE, while the second one includes the iterations that are contained in tiles whose identifiers are lexicographically less than that of set TILE. Figure 2 illustrates sets TILE_GT and TILE_LT for tile T2.

For our working example, the parametrized (with respect to indices ii, jj) set TILE is represented as follows:

```
 TILE
:= {[i, j]: ii*2 <=i<= min(((ii+1)*2 - 1), 3) and jj*2 <=j<=
min((jj+1)*2-1),3) and ii>=0 and jj>=0 ,
```

where constant "2" defines the tile size, while constant "3" states for the upper bound of iteration variables i and j.

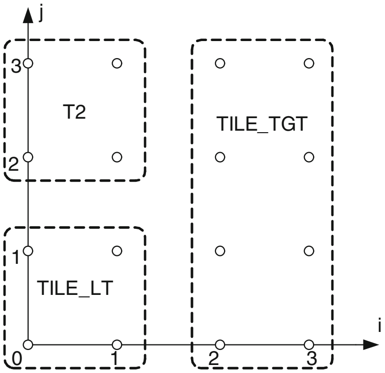


Fig. 2 Illustrating sets TILE_GT and TILE_LT

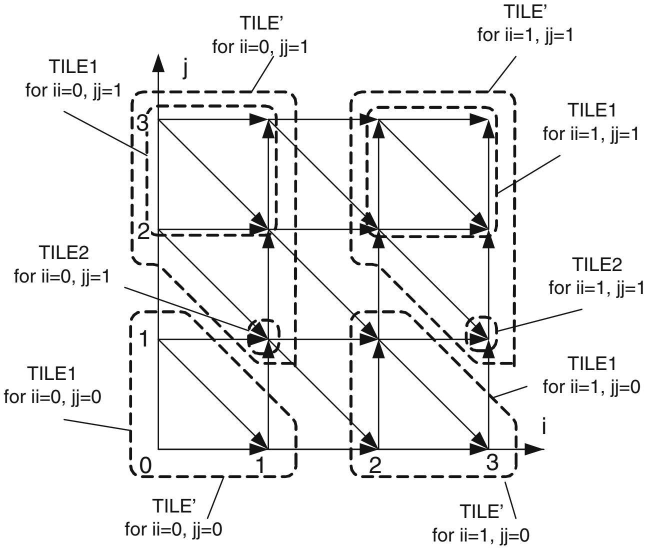


Fig. 3 Illustrating sets TILE1, TILE2, and TILE′

To calculate set, TILE1, that does not include any iteration whose execution violates a dependence(s), we remove all those iterations from set TILE that state for the dependence destinations whose sources are within tiles whose identifiers are lexicographically greater than that of TILE, by applying the following formula

TILE1 = TILE − R*(TILE_GT),

where R* is the transitive closure of the dependence graph for the working example.

The set R*(TILE_GT) includes all the iterations that are dependent on iterations belonging to set TILE_GT, while the set TILE – R*(TILE_GT) does not include any iteration that states for the destination of a dependence whose source is within set TILE_GT.

Figure 3 illustrates sets TILE1 for various values of indices ii and jj.

Now we calculate set TILE2 including all the iterations that (i) belong to tiles whose identifiers are lexicographically less than that of set TILE1, (ii) are the destinations of dependences whose sources are contained in set TILE1, and (iii) are not any destination of a dependence whose source belongs to set TILE_GT as follows

TILE2 = R*(TILE1) ∩ TILE_LT − R*(TILE_GT).

The set R*(TILE1) contains all the iterations that are dependent on the iterations belonging to set TILE1. Set TILE2 above includes iterations that are contained in tiles whose identifiers are lexicographically less than that of set TILE1 due to applying the intersection operation to the set R*(TILE1) and set TILE_LT. It is also guaranteed that the iterations belonging to set TILE2 are included only in one set because they are removed from all the tiles if they are destinations of the dependences whose sources belong to set TILE_GT. Figure 3 illustrates sets TILE2 for different values of indices ii and jj. The final formula for calculating set TILE′ is as follows TILE′ = TILE1 ∪ TILE2.

Figure 3 illustrates sets TILE′ for different values of indices ii and jj.

Due to the properties of sets TILE1 and TILE2 (they are described above), scanning tiles represented with set TILE′ (for different values of ii and jj) and iterations in each tile in the lexicographic order is valid, i.e., it is guaranteed that for each dependence, the source of a dependence is executed earlier than its destination.

Taking into account the fact that for the working example, set S, including the values of indices ii and jj defining the identifiers of tiles, is as follows

```
S := {[ii,jj]: ii >=0 and jj >=0 and ii*2 <3 and jj*2 <3 },
```

we form set TILE″ to be used for producing tiled code by means of inserting in the first positions of the tuple of set TILE′ indices ii and jj and applying correspondent constraints on them.

Applying Barvinok, for our working example, we get the following set TILE″

```
TILE″ := { [ii,jj,i,j] : (i>=2ii and i>=0 and i<=3 and i<=1+2ii
and j>=2jj and j>=0 and j<=3 and j<=1+2jj and ii>=0 and jj>=0 and
j<=1+2ii+2jj-i) or (i>=2ii and i>=0 and i<=3 and i<=1+2ii and
j>=2jj and j>=2+2ii+2jj-i and j>=0 and j<=3 and j<=1+2jj and ii>=0
and j>=1); [ii, jj, 1 + 2ii, 1] : jj = 1 and ii <= 1 and ii >= 0}.
```

Applying CLooG to set TILE″, we get the following tiled code

```
for (int c0 = 0; c0 <= 1; c0 += 1)
  for (int c1 = 0; c1 <= 1; c1 += 1)
    for (int c2 = 2 * c0; c2 <= 2 * c0 + 1; c2 += 1) {
      if (c2 == 2 * c0 + 1 and c1 == 1)
        a[c2][1]=a[c2][1+1]+a[c2][1]+a[c2][1-1];
      for (int c3 = 2*c1; c3 <= 2*c0 + 2 * c1 - c2 + 1; c3 += 1)
        a[c2][c3]=a[c2][c3+1]+a[c2+1][c3]+a[c2+1][c3-1];
      if (c2 == 2 * c0 + 1 and c1 == 1)
        a[c2][3]=a[c2][3+1]+a[c2][3]+a[c2][3-1];
    }
```

Below, we present a formal algorithm, implementing the presented idea above and allowing for the tiling transformation of the perfect loop nest of depth *d*.

It is worth to note that Algorithm 1 produces target tiles represented with TILE′whose shapes in general are different from rectangular shapes of original tiles represented with TILE. Such shapes are created automatically.

4 Related Work

There has been a considerable amount of research into tiling demonstrating how to aggregate a set of loop iterations into tiles with each tile as an atomic macro statement, from the pioneer paper [8] to those presenting advanced techniques [6, 7, 19].

One of the most advanced reordering transformation frameworks is based on the polyhedral model. Let us remind that "Restructuring programs using the polyhedral model is a three steps framework. First, the Program Analysis phase aims at translating high level codes to their polyhedral representation and to provide data dependence analysis based on this representation. Second, some optimizing or parallelizing algorithm uses the analysis to restructure the programs in the polyhedral model. This is the Program Transformation step. Lastly, the Code Generation step returns back from the polyhedral representation to a high level program" [3].

All the above three steps are available in the approach presented in this paper. But there exists the following difference in step 2: in the polyhedral model "a (sequence of) program transformation(s) is represented by a set of affine functions, one for each statement" [3] while the presented approach does not find and use any affine function. It applies the transitive closure of the program dependence graph to specific subspaces of the source loop iteration space. At this point of view, the program transformation step is rather within the Iteration Space Slicing Framework introduced by Pugh and Rosser [14], where the key step is calculating the transitive closure of the program dependence graph.

As far as perfectly nested loops are concerned, papers [8, 18] are a seminal work presenting the theory of tiling techniques based on affine transformations. These papers present techniques consisting of two steps: they first transform the original loop nest into a fully permutable one, then transform the fully permutable loop nest into tiled code. Loop nests are fully permutable if they can be permuted arbitrarily without altering the semantics of the source program. If a loop nest is fully permutable, it is sufficient to apply a tiling transformation to this loop nest [18].

Papers [2, 5] demonstrate how we can extract coarse- and fine-grained parallelism applying different Iteration Space Slicing algorithms, however they do not consider any tiling transformation.

Wonnacott and Strout review implemented and proposed techniques for tiling dense array codes in an attempt to determine whether or not the techniques permit on scalability. They write [19]: "No implementation was ever released for iteration space slicing". This permits us to state that TRACO [4], which implements the algorithm

Algorithm 1: Tiling transformation for the perfect loop nest

Input: A perfect loop nest of depth d; constants $b_1, b_2, ..., b_d$ defining the size of a rectangular original tile.
Output: Tiled code.
Method:

1. Form the following vectors, matrix, and set:
   vector $\boldsymbol{I}$ whose elements are original loop nest indices $i_1, i_2, ..., i_d$;
   vector $\boldsymbol{II}$ whose elements $ii_1, ii_2, ..., ii_d$ define the identifier of a tile;
   vectors $\boldsymbol{LB}$ and $\boldsymbol{UB}$ whose elements are lower $lb_1, ..., lb_d$ and upper $ub_1, ..., ub_d$ bounds of indices $i_1, i_2, ..., i_d$ of the original loop nest, respectively;
   vector **1** whose all d elements are equal to the value 1;
   vector **0** whose all d elements are equal to the value 0;
   diagonal matrix $\boldsymbol{B}$ whose diagonal elements are constants $b_1, b_2, ..., b_d$ defining a rectangular tile size.
2. Carry out a dependence analysis to produce a set of relations describing all the dependences in the source loop.
3. Calculate the transitive closure, R^*, of the union of all the relations extracted in step 2 applying any known algorithm, for example, that presented in [17].
4. Form set *TILE(**II**, **B**)* including iterations belonging to the parametrized tile defined with parameters $ii_1, ii_2, ..., ii_d$ as follows
   $TILE(\boldsymbol{II}, \boldsymbol{B}) = \{[\boldsymbol{I}] \mid \boldsymbol{B}*\boldsymbol{II} + \boldsymbol{LB} \leq \boldsymbol{I} \leq \min(\boldsymbol{B}*(\boldsymbol{II} + \boldsymbol{1}) + \boldsymbol{LB} - \boldsymbol{1}, \boldsymbol{UB}) \text{ AND } \boldsymbol{II} \geq 0\}$.
5. Form set *II_SET* including the identifiers of all tiles:
   $II\_SET = \{[\boldsymbol{II}] \mid \boldsymbol{II} \geq 0 \text{ and } \boldsymbol{B}*\boldsymbol{II} + \boldsymbol{LB} \leq \boldsymbol{UB}\}$.
6. Form set *TILE_LT* as the union of all the tiles whose identifiers are lexicographically less than that of TILE(**II**, **B**) as follows
   $TILE\_LT = \{[\boldsymbol{I}] \mid \text{exists } \boldsymbol{II}' \text{ s. t. } \boldsymbol{II}' \prec \boldsymbol{II} \text{ AND } \boldsymbol{II}, \boldsymbol{II}' \text{ in } II\_SET \text{ AND } \boldsymbol{I} \text{ in } TILE(\boldsymbol{II}', \boldsymbol{B})\}$,
   where $\boldsymbol{I}$ in $TILE(\boldsymbol{II}', \boldsymbol{B})$ means that vector $\boldsymbol{I}$ belongs to set $TILE(\boldsymbol{II}', \boldsymbol{B})$ being defined in step 4.
7. Form set *TILE_GT* as the union of all the tiles whose identifiers are lexicographically greater than that of TILE(**II**, **B**) as follows
   $TILE\_GT = \{[\boldsymbol{I}] \mid \text{exists } \boldsymbol{II}' \text{ s. t. } \boldsymbol{II}' \succ \boldsymbol{II} \text{ AND } \boldsymbol{II}, \boldsymbol{II}' \text{ in } II\_SET \text{ AND } \boldsymbol{I} \text{ in } TILE(\boldsymbol{II}', \boldsymbol{B})\}$,
   where $\boldsymbol{I}$ in $TILE(\boldsymbol{II}', \boldsymbol{B})$ means that vector $\boldsymbol{I}$ belongs to set $TILE(\boldsymbol{II}', \boldsymbol{B})$ being defined in step 4.
8. Form set $TILE'$ as follows
   TILE1 = TILE - R(TILE_GT), /* set TILE1 does not include the iterations that are the destinations of the dependences whose sources belong to the tiles with the identifiers that are lexicographically greater than that of TILE */*
   TILE2 = (R(TILE1) ∩ TILE_LT) - R*(TILE_GT), /* TILE2 includes all the iterations that (i) belong to the tiles whose identifiers are lexicographically less than that of set TILE1, (ii) are the destinations of dependences whose sources are contained in set TILE1, and (iii) are not any target of a dependence whose source belongs to set TILE_GT */*
   $TILE' = TILE1 \cup TILE2$.
9. Form set $TILE''$ by means of inserting i) in the first positions of the tuple of set $TILE'$ indices $ii_1, ii_2, ..., ii_d$; ii) into the constraints of set $TILE'$ the constraints defining tile identifiers
   $\boldsymbol{II} \geq 0$ and $\boldsymbol{B}*\boldsymbol{II} + \boldsymbol{LB} \leq \boldsymbol{UB}$.
10. Generate tiled code by means of applying any code generator scanning elements of set $TILE''$ in the lexicographic order, for example, CLooG [1].

presented in this paper is the first compiler where Iteration Spase Slicing is applied to produce tiled code.

Summing up, we may conclude that the algorithm presented in this paper is the first attempt to demonstrate how Iteration Space Slicing (instead of a set of affine functions, one for each statement) can be used to restructure program loops in the program transformation step of the polyhedral model to produce valid tiled code.

5 Experimental Study

The presented algorithm has been implemented in the optimizing compiler TRACO, public available at the website http://traco.sourceforge.net. TRACO includes Petit, a dependence analyzer [9], preprocessor converting output returned with Petit to a format acceptable with the Barvinok tool [16]. TRACO uses this tool to apply operations on sets and relations required by the presented algorithm. The result (sets representing tiles) is passed to the CLooG code generator and finally a postprocessor produces compilable code in C/C++. For carrying out experiments, the ISL function `isl_map_transitive_closure` [15] has been used for calculating transitive closure.

To evaluate the effectiveness of the presented algorithms, we have experimented with the NAS Parallel Benchmarks 3.2 (NPB) [12] and Polyhedral Benchmarks (PolyBench) [13].

The results in Table 1 demonstrate the effectiveness of the presented algorithm as well as that of algorithms implemented in PLUTO (version 0.9.0), an optimizing compiler [6] permitting for producing tiled code on the basis of affine transformations taking into account applying the fusion and SCCs graph splitting techniques when appropriate.

Petit is able to analyze 257 NPB loops, and dependences are available in 134 loops only (the rest 123 loops do not expose any dependence, hence producing tiled code for any of them is trivial). There exist 60 NPB perfectly nested loops, and 41 of them are double or more nested. PolyBench includes 48 loops and each one exposes dependences. There exist 14 perfectly nested loops in PolyBench and 13 of them are double or more nested. We have qualified to experiments only loops of depth 2 or more.

To study the time complexity of the approach and the efficiency of produced tiled code, we have chosen eight computatively heavy benchmarks, presented in Table 2.

Table 1 Numbers of NPB and PolyBench loops transformed with TRACO and PLUTO

Benchmark	All	Perfectly nested	Min. double nested	TRACO	PLUTO	D-form loops
NAS	134	60	41	41	20	21
PolyBench	48	14	13	13	13	0

Table 2 Code transformation times (in seconds) for PLUTO and TRACO as well as code identity

Bench	Loop(s)	Description	PLUTO	TRACO	Code identity
PolyBench	Gemver	A general matrix vector multiplication and matrix addition kernel	0.127	0.387	Yes
	Syr2k	An algorithm for symmetric rank-2k operations	0.114	0.365	Yes
	Mvt	A matrix vector product and transpose kernel	0.272	0.262	No (1)
	Seidel-2d	A 2D Seidel stencil and computation	0.458	1.002	No (2)
NPB	FT_auxfnct_2	A fast Fourier transform	0.205	0.318	No (3)
	BT_rhs_1	A block tridiagonal benchmark	0.513*	1.009	No (4)
	SP_nivr_1	A scalar pentadiagonal transform	1.469*	0.965	No (4)
	UA_setup_16	An unstructured adaptive benchmark	0.154	0.287	No (3)

By analyzing the results in Table 2, we may state that PLUTO is faster than TRACO; however, it is not able to produce code for the: *BT_rhs_1* and *SP_nivr_1* loops (they are marked with "*" in Table 2).

In general, tiled codes produced with TRACO and PLUTO are different (see the last column of Table 2).We have specified the four kinds of differences in tiled code: (1) PLUTO combines tiling and fusion techniques producing a single nest for several input nests while TRACO produces separated nests; (2) TRACO produces two tiled loops while PLUTO does three tiled loops, but experiments demonstrate very similar speed-up of both tiled codes; (3) both TRACO and PLUTO produce tiled code for all loops, but PLUTO additionally permutes tiled loops to improve locality; (4) PLUTO fails to produce any tiled code.

To evaluate the performance of produced tiled code with the presented algorithm, we have used a computer with an Intel Xeon E7310 1.6 GHz processor, 2 MB cache and 16 GB RAM. Programs have been compiled with the GNU Compiler Collection 4.3.2 and—O3 optimization.

The size of a tile has greater impact on NPB loops than on PolyBench/C ones. Figure 4 presents tiled loop speed-up in a graphical way. For all tiled loops, we observe positive speed-up (>1) that is calculated as the ratio of the execution times of original and tiled loops.

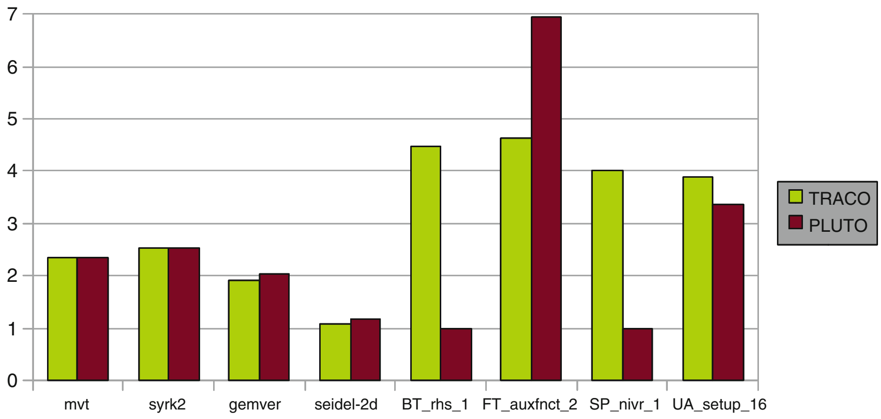


Fig. 4 Speed-up of tiled loops

6 Conclusion

In this paper, we presented a novel approach based on a combination of the polyhedral and Iteration Space Slicing frameworks permitting for tiling perfectly nested loop nests. The main merit of the presented approach in comparison with well-known ones is an increased effectiveness (the larger scope of applicability). In the future, we plan to present a modified approach to apply it for arbitrarily nested loops.

References

1. Bastoul, C.: Code generation in the polyhedral model is easier than you think. In: IEEE International Conference on Parallel Architecture and Compilation Techniques PACT'13, Juan-les-Pins, pp. 7–16 September 2004
2. Beletska, A., Bielecki, W., Cohen, A., Palkowski, M., Siedlecki, K.: Coarse-grained loop parallelization: iteration space slicing vs affine transformations. Parallel Comput. **37**, 479–497 (2011)
3. Benabderrahmane, M.W., Pouchet, L.N., Cohen, A., Bastoul, C.: The polyhedral model is more widely applicable than you think. In: Proceedings of the 19th Joint European Conference on Theory and Practice of Software. International Conference on Compiler Construction, CC'10/ETAPS'10, pp. 283–303. Springer, Berlin (2010). http://dx.doi.org/10.1007/978-3-642-11970-5_16
4. Bielecki, W., Palkowski, M.: A parallelizing and optimizing compiler—traco. http://traco.sourceforge.net (2013)
5. Bielecki, W., Palkowski, M., Klimek, T.: Free scheduling for statement instances of parameterized arbitrarily nested affine loops. Parallel Comput. **38**(9), 518–532 (2012)
6. Bondhugula, U., Hartono, A., Ramanujam, J., Sadayappan, P.: A practical automatic polyhedral parallelizer and locality optimizer. SIGPLAN Not. **43**(6), 101–113 (2008)
7. Griebl, M.: Automatic parallelization of loop programs for distributed memory architectures (2004)

8. Irigoin, F., Triolet, R.: Supernode partitioning. In: Proceedings of the 15th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL'88, pp. 319–329. ACM, New York (1988)
9. Kelly, W., Maslov, V., Pugh, W., Rosser, E., Shpeisman, T., Wonnacott, D.: The Omega project. http://www.cs.umd.edu/projects/omega/release-1.0.html
10. Kelly, W., Pugh, W., Rosser, E., Shpeisman, T.: Transitive closure of infinite graphs and its applications. Int. J. Parallel Program. **24**(6), 579–598 (1996)
11. Lim, A., Cheong, G.I., Lam, M.S.: An affine partitioning algorithm to maximize parallelism and minimize communication. In: Proceedings of the 13th ACM SIGARCH International Conference on Supercomputing, pp. 228–237. ACM Press (1999)
12. NAS benchmarks suite. http://www.nas.nasa.gov (2013)
13. The Polyhedral Benchmark suite. http://www.cse.ohio-state.edu/pouchet/software/polybench/ (2012)
14. Pugh, W., Rosser, E.: Iteration space slicing and its application to communication optimization. In: International Conference on Supercomputing, pp. 221–228 (1997)
15. Verdoolaege, S.: Integer set library—manual. Technical Report http://www.kotnet.org/skimo//isl/manual.pdf (2011)
16. Verdoolaege, S.: Barvinok: User guide. Version: barvinok-0.36. http://garage.kotnet.org/skimo/barvinok/barvinok.pdf (2012)
17. Verdoolaege, S., Cohen, A., Beletska, A.: Transitive closures of affine integer tuple relations and their overapproximations. In: Proceedings of the 18th International Conference on Static Analysis, SAS'11, pp. 216–232. Springer, Berlin (2011). http://dl.acm.org/citation.cfm?id=2041552.2041570
18. Wolf, M.E., Lam, M.S.: A loop transformation theory and an algorithm to maximize parallelism. IEEE Trans. Parallel Distrib. Syst. **2**(4), 452–471 (1991)
19. Wonnacott, D.G., Strout, M.M.: On the scalability of loop tiling techniques. In: Proceedings of the 3rd International Workshop on Polyhedral Compilation Techniques (IMPACT) (January 2013)
20. Xue, J.: On tiling as a loop transformation. Parallel Process. Lett. **7**(4), 409–424 (1997)

Parallel Byzantine Fault Tolerance

Maciej Zbierski

Abstract The increasing popularity of distributed systems and applications has generated the need for algorithms guaranteeing high availability and reliability of such solutions. As an answer to this demand, various Byzantine fault-tolerant algorithms have been designed, allowing the systems to provide the correct service even in the presence of faults, either accidental or of malicious origin. However, despite significant efforts recently made, their practicality is still limited by many factors, such as the cost of additional machines or decreased system throughput. Addressing these issues, we propose Apex, a parallel Byzantine fault-tolerant execution algorithm. By processing independent requests in parallel on different multicore machines, we were able to obtain a significant increase in throughput over traditional algorithms. The performed tests have shown that our approach can execute the incoming packs of requests even several times faster than other similar algorithms.

Keywords Byzantine fault tolerance · Dependability · State machine replication · Distributed systems · Parallel processing · Scalability

1 Introduction

Over the years, fault tolerance has become an integral compound of practical distributed systems, due to an increased demand on ensuring their high availability and correctness. A particularly important aspect of this issue is the ability to tolerate Byzantine, i.e., arbitrary faults, with no assumptions about their origin nor characteristics. As opposed to classical models, Byzantine faults can lead to inconsistent effects on different destinations. This behavior can originate from design errors, environmental influence, hardware malfunction, or even an active attacker altering the system. While most research on dependable computing is targeted on classical

M. Zbierski (✉)
Institute of Computer Science,
Warsaw University of Technology, Warszawa, Poland
e-mail: m.zbierski@ii.pw.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_27

models [7, 11, and references therein], Byzantine faults still need special attention in critical real-time and distributed systems [1].

Algorithms and protocols tolerating Byzantine faults have been successfully applied in various types of critical real-time systems, for instance, inside airplanes [15]. However, contemporary solutions are still both expensive and problematic to implement. This is caused both by the high complexity of the algorithms and the additional cost required to guarantee the assumed fault model. On the other hand, the Byzantine fault-tolerant systems are very desirable, as they help to protect unpredicted errors. In the past, various incidents, originating for instance from bugs in software [8] or faulty hardware [18], have occurred, rendering traditional crash fault-tolerant services unavailable sometimes even for several hours.

We believe that in order to make the Byzantine fault tolerance practical and widely applicable, an increased fault-tolerance model is not enough. Apart from that, other benefits need to be provided, so that the cost of additional machines and everything that follows is somehow compensated.

An entry point to our proposition is an observation about the state access patterns in replicated systems. The majority of previous studies have implicitly assumed that every request might access all contents of the state (see for instance [2, 4, 12, 23]). While this is true in some situations, such as when the state size is relatively small, a significant number of applications allow to reduce the part of the state accessed in every request. An example of such situation might be replicated databases, or distributed file systems, where each request usually accesses relatively small fraction of the total state [9]. As a response to this observation, following an example of several previous solutions (for instance [5, 13]), we divide the application state into a number of smaller entities called *objects*. Similarly as before, this division is purely arbitrary and depends on the application.

In our approach, we group multiple state objects into so-called state parts and assign each part to a group of $f + 1$ machines. This allows our solution to exploit two different types of parallelism while processing the incoming requests. First, each replica executes only the requests accessing the objects grouped in its assigned state part. For instance, in a system containing two state parts, provided the state objects are accessed evenly, each replica would only process half of all the requests. Second, every replica can process independent requests in parallel, possibly on multiple processor cores.

In this article, we present an original Byzantine execution algorithm, Apex, that takes advantage of the state access parallelization described above to increase its overall efficiency. The proposed solution can be combined with the vast majority of existing agreement protocols requiring only minor modifications in their code. By maintaining lazy consistency between all the replicas, the protocol provides fast recovery in a situation where faulty machines are detected.

The article is constructed as follows: we start by describing the applied state division approach and other assumptions about the distributed system in Sect. 2. Section 3 presents the protocol for both single and multiple state objects accesses. Section 4 evaluates the algorithm and discusses results obtained from the simulation scenarios. Finally, we present the related work in Sect. 5 and conclude in Sect. 6.

2 System Model

This section presents the general system model and describes the state division approach used by the proposed execution algorithm.

2.1 *General System Model*

Our algorithm can be used to replicate a service to a number of n machines or replicas. The operations performed by the service can be arbitrary, as long as they are deterministic. For example, apart from basic read/write access, they can involve any computations performed on the state and/or operation arguments. Finally, these operations are invoked from the clients connected to a replica by issuing a request and transmitting it using the underlying network protocol.

Throughout this article, we assume the standard system model used by other Byzantine fault-tolerant protocols [2, 12, 23]. We assume that at most f out of n replicas and any number of clients can fail arbitrarily. The network interconnecting the replicas can discard, corrupt, or delay the exchanged messages. Similarly to previous algorithms in that field, our solution is safe under the asynchronous model, while the liveness is guaranteed as long as the *bounded fair links* [23] assumption holds. The full description of safety and liveness guarantees can be found in [2].

We take advantage of the solution proposed by Yin et. al. [23] and utilize different protocols for agreement and execution. In such approach, the incoming requests are handled by the agreement protocol which determines the order in which they should be processed. Afterward, the execution protocol processes the requests in the previously established order. In this article, we focus mostly on the execution phase, assuming the order of incoming requests has already been determined. The agreement phase can be realized by existing protocols (e.g., [2–4, 21, 24]) requiring only minor modifications in their code, as described in the next section. Finally, we will use the terms agreement and execution replicas whenever we refer to machines performing agreement and execution protocols respectively.

We divide the whole state replicated among the machines to a number of *objects*. For example, in terms of a database application, one object might correspond to one table. For simplicity, we additionally assume that no two objects can overlap. Let these objects be in turn arbitrarily grouped into a certain number of *parts*.

2.2 *Selective Request Execution*

Upon the service setup, the total number of replicas n is arbitrarily divided into p groups, in a way that each group contains no less than $f + 1$ machines and p denotes the number of state parts. Subsequently, each state part is assigned to a replica group so that $\forall i \in \{1 \dots p\}$ all machines in group i are responsible for executing the requests to objects located inside state part S_i. For instance, requests to objects located inside part 1 would be executed by replicas from group 1, requests to part number 2 by the second group etc. For simplicity, we assume that each group contains exactly $f + 1$ replicas, thus $n = p \times (f + 1)$.

Please note that although each replica is only responsible for executing requests to objects from the state part assigned to it, every machine still maintains a copy of the whole state and refreshes its contents upon obtaining the upgrade messages from other replicas. This is done either immediately after the reception of such message, or if the cost of updating the state is significantly high, the messages containing state updates are buffered to be used whenever either the replica is not occupied or it requires an up-to-date contents of that object.

The clients can issue requests to access (read or modify) any set of state objects. Let us denote the set of objects accessed by a request r as $AO(r)$. Additionally, let $AP(r)$ be the set of accessed parts, that is:

$$AP(r) = \{p \in parts : \exists_{o \in AO(r)}\ p \text{ contains } o\}. \tag{1}$$

The sequence of execution is enforced by the agreement protocol, which assigns each request a unique identifier in an increasing order. Apart from assigning that value, for every accessed state object, the agreement protocol attaches an identifier of the last request to that object. We introduce a predicate $prev(r, o)$, which for a request r returns the identifier of the previous task performing operations on object o. Such values suffice to preserve the total ordering of requests.

3 The Specification of Apex Protocol

In the fully replicated execution protocols each replica performs the requests in the same order established during the agreement phase. Contrary to that approach, for every request each Apex replica takes exactly one of the available roles and becomes either an executor, verifier, or an observer. The executors perform the requests and send their results to the clients. This behavior is similar to the one of machines in traditional execution algorithms. The verifiers on the other hand do not execute the requests, but rather ensure that all executors perform their tasks and generate the same result. Finally, the observers do not take an active role in the execution process, and only update their state as soon as they receive a result from other replicas.

3.1 Accessing Single State Part

In this section, we assume that while the client might issue a request to any nonempty set of objects, all of them belong to the same part. In terms of the introduced nomenclature, this is equivalent to $|AO(r)| \geq 1$ while $|AP(r)| = 1$ for every incoming request r.

As mentioned before, the roles are assigned separately for every request, based on the location of the state objects being accessed. Whenever a client issues a request to access object o assigned to the ith part, replicas from group number i will assume the role of executors. Replicas from the next group (or from the first one, if an object from the last state part is being accessed) will on the other hand perform the role of verifiers. Finally, machines from other groups will take the role of observers. The verifier role is assigned only for single state part accesses; we will discuss this further in Sect. 3.2.

Let us introduce a predicate $completed\_globally(r)$, which is true for a given replica when it has obtained the same result to the request r from at least $f + 1$ different executors (possibly including itself), and false otherwise. Please note that these results could have been either transmitted directly by the executors or forwarded by verifiers. Additionally, we define a predicate $completed\_locally(r)$, which is true for a certain replica if either $completed\_globally(r)$ is true for that replica or it has already executed request r by itself.

Algorithm 1 Preparing for request execution.

```
Upon reception of request $r$
if replica_role($r$, $me$) = EXECUTOR then
    execute_request($r$)
end if

Upon reception of $response$ to $r$
if replica_role($r$, $me$) = VERIFIER then
    $resp\_count$ := #(identical responses to $r$)
    if $resp\_count = 1$ then
        set timeout on $r$
    else if $resp\_count = f + 1$ then
        cancel timeout on $r$
    end if
    $sender\_role$ := replica_role($r$, $response.from$)
    if $sender\_role$ = EXECUTOR then
        send $response$ to other replicas and client
    end if
end if

Upon timeout on $r$ or contradictory results to $r$
if replica_role($r$, $me$) = VERIFIER then
    cancel timeout on $r$
    execute_request($r$)
else if replica_role($r$, $me$) = OBSERVER then
    notify verifiers
end if
```

After receiving the new request r, machines behave differently based on their role for that request. Executors process the request as soon as the following statement is satisfied:

$$\forall o \in AO(r)\ completed\_locally(prev(r, o)). \tag{2}$$

As soon as r has been processed, the executors sign the obtained request and relay it to the client and all the other replicas. Verifiers do not execute the obtained task, but instead wait for responses from the executors. There will be at least one such message for every client query, since the number of executors is always greater than the total number of possibly faulty machines f. Verifiers relay the obtained message to other replicas, with an exception that for every request a result from each executor is forwarded only once.

Replicas that do not execute the queries themselves (i.e., verifiers and observers) can apply the result of a request r to their copy of the state as soon as $completed\_globally(r)$ becomes true, provided the result of a request with a higher identifier has not already been applied.

Algorithm 2 Executing the requests.

```
Procedure execute_request($r$)
for $object \in AO(r)$ do
    wait until $completed\_locally(prev(r, object))$
end for
$result$ := r.execute()
send $result$ to other replicas and client
```

It is worth noting that with only $f + 1$ processing replicas, for an arbitrary request r, $completed\_globally(r)$ will never be true if at least one of those machines crashes or produces a different result. This is caused by the fact that the required minimum number of the identical responses is equal to the number of executors, i.e., $f + 1$. To circumvent this, verifiers set a timer after receiving the first response to every request. If a verifier does not receive enough responses for $completed\_globally(r)$ to be true before the timeout, it temporarily takes over the role of an executor for that request. Such replicas begin to process the job as soon as the objects accessed by the query are updated to their latest values.

The pseudocode for the process described above has been presented as algorithm 1. The first part includes the behavior of the executor after receiving a new request (lines 1–3). The second considers the possible actions undertaken by the verifiers after obtaining a result of a request. These include starting and stopping the response timer (lines 5–10) and relaying the response, if it was received directly from an executor (lines 11–14). Finally, the last part presents the behavior of replicas when faulty machines are detected, including the re-execution of the request by the verifiers (lines 17–18) either because of a timeout or after obtaining the contradictory results. The function for performing the requests ($execute\_request$) will be described in the next section.

3.2 Multiple State Parts Access

The situation where a client accesses objects located inside multiple state parts is somewhat simpler than the single part access. In such case the request is executed by at least two replica groups, i.e., at least $2f + 2$ machines. As described before, this is more than enough to guarantee at least $f + 1$ identical results. Consequently, with multiple state parts accesses no verifiers are needed. In other words, replica from group g performs the role of an executor for the request r iff. $g \in AP(r)$; otherwise it performs the role of an observer.

Similarly as for single state part tasks, before executing the request r, executors need to wait until $completed\_locally(r)$ becomes true. This is automatically satisfied when that machine has processed every prerequisite of the request r. Otherwise, it might need to wait for a response from other executors.

The pseudocode for the request execution function is presented as Algorithm 2. Lines 1–3 are responsible for ensuring the correct initial state by satisfying Eq. 2. The rest of the code executes the request (line 4) and broadcasts the result (line 5). Please note that this algorithm is the same both for single and multiple state parts requests.

4 Experiments and Evaluation

In this section, we evaluate our approach by discussing the results of various experiments involving Apex and two other modern Byzantine replication protocols: PBFT [2] and ODRC [5]. PBFT is usually treated as a baseline protocol in such comparisons, while ODRC is a solution that, similarly to Apex, takes advantage of selective request execution.

Although at the first glance Apex might appear very similar to ODRC, since both execute independent requests on a subset of replicas, their construction differs substantially. First of all, in ODRC each replica group processes exactly one sequence of requests. Namely, each replica is equipped with exactly one queue for requests accessing the objects maintained by that replica. Whenever new tasks arrive, they are inserted into the queue. Similarly, the workers remove elements from the queue as soon as it is nonempty and the previous request has been processed. The solution proposed by Apex on the other hand, although in reality constructed in a slightly different manner, can be thought of as a set of queues, each containing requests accessing a certain maintained object. Consequently, Apex would process all requests to different objects located on the same replica in parallel, while ODRC would execute them sequentially.

Another main difference between Apex and ODRC appears in the role of the clients. In ODRC, apart from issuing the requests, the clients are also responsible for the detection of possibly faulty replicas. Consequently, rogue clients, while not posing a threat to safety, can slow down the whole system by enforcing all replicas to process every request by simply claiming they have not received enough responses. In Apex, the detection of faulty replicas is done by the verifiers on the server side, while the clients can only inform them if they have not yet received a response.

Finally, the two protocols provide different consistency models. Apex maintains consistency between all the machines, while in ODRC only objects managed by the replica are kept up to date. This might lead to an increased duration of ODRC's recovery procedure, although will inevitably be faster in situations where the execution time of a single task is of the same order of magnitude as the time required to write the state value. However, since both ODRC and Apex can be modified to support each model, the decision which approach is better should be made based on the expected use cases of the implemented application.

4.1 Test Environment

In order to evaluate our approach we have performed a series of tests simulating the behavior of our algorithm in different situations. We have used the Neko framework [20] to provide an implementation of all the protocols. The simulations were performed using a group of servers connected to the same local area network.

For the purpose of our experiments, we have divided the whole state into 12 objects and grouped them into three parts, each consisting of four objects. This choice has been made based on the number of available processors in every machine. The sample setting consisting of total 6 machines allowed to tolerate at most $f = 1$ faulty node. While this setting uses 2 more machines than a PBFT system also tolerating one faulty node, we will show that at such relatively low cost the requests can be processed even several times faster.

The execution of a request has been simulated by performing arbitrary computations for a certain amount of time specified upon its creation. We have divided the incoming requests into two main categories: short- and long-lasting. The first group simulates simple read/write operations, such as database accesses. The second one, on the other hand, represents more demanding computations. Job durations were generated using a normal distribution with the mean equal to $\mu_{\text{short}} = 10\,\text{ms}$, $\mu_{\text{long}} = 1{,}000\,\text{ms}$ and standard deviation of $\sigma_{\text{short}} = 5\,\text{ms}$, $\sigma_{\text{long}} = 500\,\text{ms}$ respectively. Unless stated otherwise, we have performed 20 trials of every experiment and present the mean of obtained results.

As we have stated before, the strength of Apex lies in the possibility of parallelizing state access operations. As a result, its efficiency will vary based on the choice and order of state objects accessed by incoming client requests. To verify this, we have selected two contrarily different access patterns. In the optimistic pattern, the client requests are divided more or less evenly among all objects, whether in the pessimistic one every client accesses the same object.

These patterns have been used in the tests to simulate client traffic in the comparison between Apex, PBFT, and ODRC. The ODRC has been tested for the optimistic pattern only, since both Apex and ODRC perform very similarly for the pessimistic

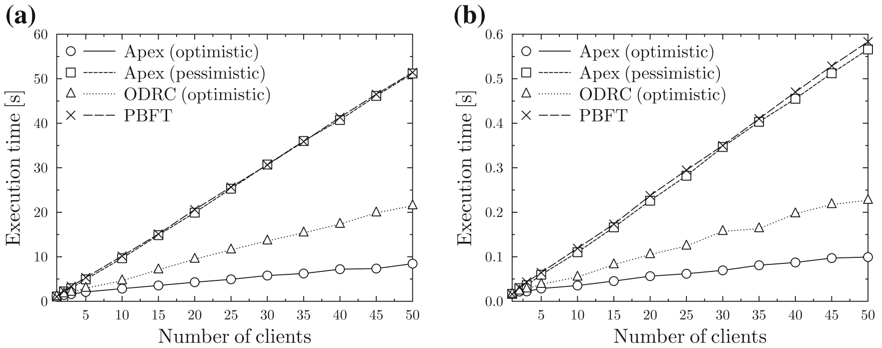


Fig. 1 Total execution times for **a** long requests ($\mu = 1\,\mathrm{s}$, $\sigma = 0.5\,\mathrm{s}$) and **b** short requests ($\mu = 10\,\mathrm{ms}$, $\sigma = 5\,\mathrm{ms}$)

case. Furthermore, since PBFT performs the same regardless the choice of state access pattern, it was executed and measured only for the optimistic scheme.

4.2 Normal Case Operation

Figure 1 presents the results for different access patterns, for both long and short request processing time. The obtained results demonstrate that for the optimistic access pattern Apex has processed the same set of requests around six times faster than PBFT in case of long tasks and nearly as much for shorter ones. Similarly, the gain over ODRC oscillates around 2.5 times for both request types. Although the reduction is not as high as the theoretical maximum (12 and 4 times respectively), this can be explained by the fact that the requests have been assigned to the state objects at random, but not necessarily evenly. Finally, the efficiency of the pessimistic pattern is very similar to the results obtained by PBFT.

4.3 Introducing Faults

Introducing Byzantine faults to our model results in a broad variety of additional possible actions that can be undertaken by the faulty replicas. However, since the protocol is both safe and live under the previously stated assumptions, the most significant possible consequence of those actions is the increase in the total processing time, which originates from the need to launch additional execution on verifiers' machines. Consequently, in order to achieve this, faulty machines can either produce incorrect results or generate no response at all. Please note however that no other

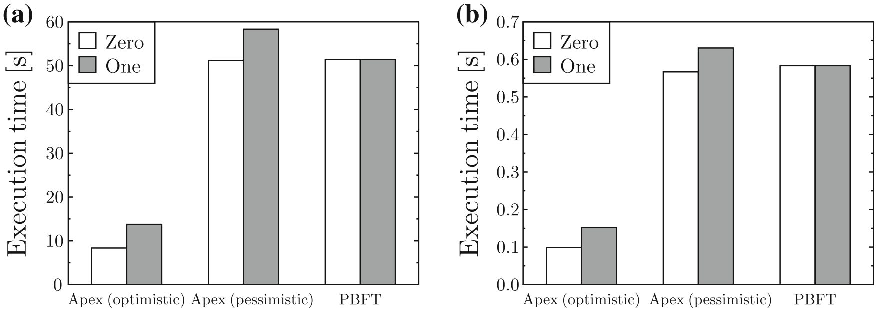


Fig. 2 Execution times in environments with zero or one faulty replica for **a** long requests ($\mu = 1$ s, $\sigma = 0.5$ s) and **b** short requests ($\mu = 10$ ms, $\sigma = 5$ ms)

action performed by the replicas, such as for instance equivocation, should lead to a greater decrease in the efficiency of the algorithm than failures by crashing.

In all the experiments in this section, we assume that the pessimistic pattern represents accesses to an object located inside the state part associated with the replica group containing the faulty machine. Additionally, we assume that replicas will treat other machines as faulty when they do not obtain an expected response for 5 s and 50 ms for long and short requests, respectively (five times the nominal request execution time). It is worth noting that while the results presented below will vary depending on the selected timeout, we believe that the value we have chosen is sufficient in most applications.

Figure 2 presents the difference in execution times for 100 clients using various state part access patterns in a situation where either all machines are correct or exactly one of them has failed by crashing. While the faulty machine has increased the overall processing time in the optimistic pattern by around 60 %, the obtained results are still better than the one of PBFT. Additionally, although the execution time for the pessimistic scenario with one faulty replica is longer than for PBFT, the difference between them oscillates around 10 %.

5 Related Work

The foundation of modern Byzantine fault-tolerant algorithms [16] is PBFT by Castro and Liskov [2], which has been used in this article as a reference algorithm. The concept of executing the requests only on a subset of processors has originally been presented in [19]. The article distinguishes two types of replicas: copies and witnesses. Copies execute the requests and store the full contents of the state. Witnesses, on the other hand, keep only the value of a counter. Whenever a write query is performed, copies execute it and witnesses just increase their counter. This proposition might lead to storage space optimization; sometimes also reading the value of the counter might be faster than fetching the whole state. This approach was later reused in [10].

In the Harp file system [17] the witnesses are used only during the recovery protocol (view change) to form the required quorum. Ladin et. al. [14] have shown that executing requests only on a subset of replicas enables to develop much more efficient crash-tolerant algorithms. The proposed approach uses lazy replication, i.e., one where the contents of the state are updated by a periodic exchange of "gossip" messages between executors and witnesses.

Certain systems like SPARE [6] or ZZ [22] aim to minimize the number of concurrently operating replicas. During gracious executions both algorithms require only $f+1$ machines, and are able to activate up to f more when faults are detected. While this approach is understandable in a situation where the reduction of execution cost is critical, we have shown that additional replicas can be used to increase the total throughput of the system.

Parallel processing of independent requests has previously been used by Kotla et. al. in CBASE [13]. The described approach introduces an additional layer between the agreement and execution to determine possibly independent tasks, such as queries to nonoverlapping state objects. Such requests are then processed by different worker threads. CBASE does not however divide replicas into groups, and as a result every request is still executed on every machine.

Hendricks et. al. [9] have introduced the notion of Byzantine Locking based on a lock service deployed atop a standard replication algorithm. The proposed protocol grants users an ability to obtain an exclusive access to a portion of the state, while the underlying algorithm is used for service actions, e.g., state extraction and integration. However, the Byzantine Locking assumes that state objects are rarely shared between the requests, as in such cases its efficiency is determined by that of the underlying replication algorithm, such as PBFT [2] or Zyzzyva [12].

6 Conclusion

In this article we have presented Apex, an original algorithm for handling requests in Byzantine fault-tolerant distributed systems. By applying a high level of parallelism, our solution is more suitable for multicore environments than the previously presented approaches. We have shown that Apex can obtain a significant speedup of even several times over PBFT, the baseline algorithm in that field, and ODRC, a modern approach leveraging selective execution. Additionally, we have demonstrated that our solution performs well even for unfavorable cases, that is in the presence of faulty replicas. We believe that these concepts enable Apex to be used as a building block for services that could significantly outperform existing Byzantine fault-tolerant solutions.

In the future, we plan to enhance our simulations by implementing a real-life service based on Apex. Additionally, we intend to continue our work on Byzantine fault-tolerant algorithms and extend their scope to geographically distributed environments, characterized by high transmission latency and varying bandwidth.

References

1. Avizienis, A., Laprie, J.-C., Randell, B., Landwehr, C.: Basic concepts and taxonomy of dependable and secure computing. IEEE Trans. Dependable Secur. Comput. **1**(1), 11–33 (2004)
2. Castro, M., Liskov, B.: Practical Byzantine fault tolerance. In: Proceedings of the Third Symposium on Operating Systems Design and Implementation, OSDI'99. USENIX Association, Berkeley, pp. 173–186 (1999)
3. Chun, B.-C., Maniatis, P., Shenker, S., Kubiatowicz, J.: Attested append-only memory: making adversaries stick to their word. In: Proceedings of the 21st Symposium on Operating Systems Principles (2007)
4. Correia, M., Neves, N.F., Veríssimo, P.: How to tolerate half less one Byzantine nodes in practical distributed systems. In: Proceedings of the 23rd IEEE Symposium on Reliable Distributed Systems, pp. 174–183, October 2004
5. Distler, T., Kapitza, R.: Increasing performance in Byzantine fault-tolerant systems with on-demand replica consistency. In: Proceedings of the EuroSys 2011 Conference (EuroSys'11), pp. 91–105 (2011)
6. Distler, T., Kapitza, R., Popov, I., Reiser, H.P., Schröder-Preikschat, W.: SPARE: replicas on hold. In: Proceedings of the 18th Network and Distributed System Security Symposium (NDSS '11), pp. 407–420, (2011)
7. Gawkowski, P., Sosnowski, J.: Dependability evaluation with fault injection experiments. IEICE Trans. Inf. Syst. **E86-D**(12), 2642–2649 (2003)
8. Google. App engine outage today, p. 6 (2008). https://groups.google.com/forum/?fromgroups=#!topic/google-appengine/985VmzuLMDs
9. Hendricks, J., Sinnamohideen, S., Ganger, G.R., Reiter, M.K.: Zzyzx: Scalable fault tolerance through byzantine locking. In: 2010 IEEE/IFIP International Conference on, Dependable Systems and Networks (DSN), pp. 363–372 (2010)
10. Huang, A.C., Fox, A.: Cheap recovery: a key to self-managing state. ACM Trans. Storage **1**(1), 38–70 (2005)
11. Knight, J.: Fundamentals of Dependable Computing for Software Engineers. Boca Raton, CRC Press, (2012)
12. Kotla, R., Clement, A., Wong, E., Alvisi, L., Dahlin, M.: Zyzzyva: speculative Byzantine fault tolerance. In: Symposium on Operating Systems Principles (SOSP) (2007)
13. Kotla, R., Dahlin, M.: High throughput Byzantine fault tolerance. In: Proceedings of the 2004 Conference on Dependable Systems and Networks, pp. 575–584 (2004)
14. Ladin, R., Liskov, B., Shrira, L., Ghemawat, S.: Providing high availability using lazy replication. ACM Trans. Comput. Syst. **10**(4), 360–391 (1992)
15. Lamport, L.: From Byzantine generals to hackers, p. 5 (2011), http://www.college-de-france.fr/site/martin-abadi/seminar-2011-05-11-11h00-resume.htm
16. Lamport, L., Shostak, R., Pease, M.: The Byzantine generals problem. ACM Trans. Progra. Lang. Syst. **4**, 382–401 (1982)
17. Liskov, B., Ghemawat, S., Gruber, R., Johnson, P., Shrira, L.: Replication in the Harp file system. In: Proceedings of the Thirteenth ACM Symposium on Operating Systems Principles, SOSP '91, pp. 226–238, ACM. New York (1991)
18. Osier, M.: Shipping delay recap, p. 8 (2008), http://blog.netflix.com/2008/08/shipping-delay-recap.html
19. Pâris, J.-F.: Voting with witnesses: a consistency scheme for replicated files. In: Proceedings of the 6th International Conference on Distributed Computing Systems, pp. 3–6 (1986)
20. Urbán, P., Défago, X., Schiper, A.: Neko: A single environment to simulate and prototype distributed algorithms. J. Inf. Sci. Eng. **18**(6), 981–997 (2002)
21. Veronese, G.S., Correia, M., Bessani, A.N., Lung, L.C., Verissimo, P., Efficient Byzantine fault tolerance. IEEE Trans. Comput. p. 99 (2011)
22. Wood, T., Singh, R., Venkataramani, A., Shenoy, P., Cecchet, E.: ZZ and the art of practical BTF execution. In: Proceedings of the sixth conference on Computer systems, EuroSys'11, pp. 123–138 (2011)

23. Yin, J., Martin, J.-P., Venkataramani, A., Alvisi, L., Dahlin, M.: Separating agreement from execution for Byzantine fault tolerant services. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, pp. 253–267, ACM Press (2003)
24. Zbierski, M., Iwazaru: The Byzantine sequencer. In: Proceedings of the 26th International Conference on Architecture of Computing Systems, pp. 38–49 (2013)

Increasing Web Services Discovery Relevancy in the Multi-ontological Environment

Larysa Globa, Mykhailo Kovalskyi and Oleksandr Stryzhak

Abstract The existing approaches to the solution of Web services discovery issues are reviewed briefly in this paper. The problems of low relevancy, interoperability, and high amount of the human factor existing in the process of Web services discovery are found; these can be solved by addition of semantic data to Web service description. The existing models of ontology-annotated Web services are analyzed and their modifications allowing to achieve higher discovery relevancy are proposed.

Keywords Web service · Ontology · Relevancy

1 Introduction

Web services provide unified APIs, which allow their usage by developers and end-users, in case the latter ones require the capabilities of Web services.

Such model of Web service-based distributed computing system should be found that end-users were able to discover required Web services matching search criteria automatically with the highest possible relevancy. It is still a problem, e.g., UDDI (Universal Description, Discovery, and Integration) registry provides a Web services

L. Globa (✉) · M. Kovalskyi
National Technical University of Ukraine "Kiev Polytechnic Institute", Kiev, Ukraine
e-mail: lgloba@its.kpi.ua

M. Kovalskyi
e-mail: mkovalskyi@luxoft.com

O. Stryzhak
Institute of Telecommunications and Global Informational Environment,
National Academy of Science of Ukraine, Kiev, Ukraine
e-mail: sae953@gmail.com

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_28

lookup only using keywords and predefined categories in which target Web services may be located. Therefore, a final decision is left to end-user.

A lot of works suggest different solutions for this problem. There is a semantic Web services approach among them. This solution is based on supplementing the existing WSDL (Web Services Description Language) descriptions by annotating the latter ones with ontological models, which can be developed separately from Web services descriptions. However, these existing methods are characterized by low relevancy of results containing discovered Web services that makes theirs usage more difficult for end-users.

2 Analysis of the Existing Approaches

By the fact, the standard of Web services discovery is UDDI registry, which provides a categorized search by the keywords. The variety of researchers consider this approach as nonscalable, because it requires a full replication of the Web services published in the registry [1]. For the purpose of solving this problem, there are approaches of the "peer-to-peer" ideology proposed [2]. These include solutions of distributively stored set of the Web services descriptions with partial or full replication.

There was an effort to analyze the possibility of process taxonomy, describing allowed actions with entity, usage regarding this issue [3]. Such method uses semantic relations between processes to reflect similarities and differences between end-user requests and models describing Web services.

Method described in [4] uses Liskov substitution principle regarding ontology model concepts. Each concept may be replaced by its sub-concept in case the latter one provides the same or wider interface in some definite context. Thus, more detailed ontological models can be adapted to be compared with less detailed ones. Among cons of this method there is problem of language-bounding, i.e., it uses Web services annotated with ontological concepts exclusively by DAML (DARPA Agent Markup Language) instruments. DAML gives a limitation in terms of annotations for Web services, because it does not support descriptions for inputs and outputs of Web services' operations. Moreover, analysis of resulting annotations is possible only by using names and descriptions of Web services; this results into raising false matching possibility.

Another proposed solution [5] stands for transformation of Web services descriptions in view of the RDF-graphs (Resource Description Framework). Next, the latter ones are compared with each other; such comparison involves decomposition of nodes representing complex types into simpler ones.

An approach to comparison of ontological models by LARKS (Language for Advertisement and Request for Knowledge Sharing) language facilities also has place [6]. The core of the system, which is described in this particular paper, consists of five distinct so-called "filters"—search criteria. Such concept of "filters" is based on approximation of ontological models representing Web services into set of keywords, but not on comparison of distinct components of ontological model. The calculated set of keywords are used next for comparison with end-user request.

LARKS was developed taking in account the specific properties of multi-ontological environment. However, in terms of LARKS it is considered that any ontological model uses some unified glossary or generalized ontological model.

In the [7] paper, there could be found the analysis of the possible ontological model mismatches depending on intents and utilities used while designing a definite model. The same research shows that statistic and linguistic algorithms can be applied to comparison of concepts' properties and it does seem reasonable.

There are also solutions in machine learning area, e.g., LSD (Learning Source Descriptions) which are based on probability distribution of the instances of the variety of the predefined ontological models.

However, in practice there is still high amount of human factor and low Web service discovery response relevancy. It forces the end-user to pick correct search result from response. The method of ontology-annotated Web services discovery proposed in [1] is based on syntactic similarity, which involves similarity of names and descriptions of concepts related to Web services. The concepts, which are engaged into comparison, are compared by using syntactic similarity, which in its turn uses different string comparison algorithms. The researchers came up with conclusion, that this method shows good results in single-ontology environment where a single-ontological model is used, but not in multi-ontological environment where multiple ontological models may be used. Based on results of research performed, authors argue that discovery response relevancy for end-user request considerably decreases for multi-ontological environment.

The analysis of the reviewed methods allows to make a conclusion about their low efficiency. However, the method proposed in [1] shows the highest relevancy for Web service discovery in single-ontological environment, thus it was chosen for the further improvements. These improvements include the following items.

First, no definite description language should be used, in this case we are trying to unbind from DAML. Used algorithm does not depend on the definite implementation of the ontological model, but it does depend on description language. A proxy abstract object called "service template" is proposed to be added, i.e., such abstraction which allows reflecting into it both end-user request and ontological models describing Web services.

Second, previously Web services similarity was calculated only by using syntactic similarity [1]. For the relevancy improvement purposes, it is proposed to include also semantic similarity, which involves inputs and outputs of the operations, e.g., taking in account the cardinality of parameter type.

3 Web Service Similarity Evaluation Method

Approach to finding a Web service in multi-ontological environment consists of three consecutive steps:

1. Creating a "service template" based on user request;

2. Comparison of “service template” with multiple Web services that were identified as candidate-services;
3. Returning Web services satisfying the minimum acceptable assessment of similarity to the user as an ordered list.

Semantic “service template” describes the user’s query. It allows the end-user to specify a set of required operations, their properties, inputs and outputs. A “service template” has no specific implementation, because it should be seen as an intermediate abstraction—proxy Web service.

More formally, a service template (ST) can be determined as follows:

$$\mathrm{ST} = \langle \mathrm{N_{ST}}, \mathrm{D_{ST}}, \mathrm{OPs_{ST}} \langle \mathrm{N_{OP}}, \mathrm{D_{OP}}, \mathrm{O_{OP}}, \mathrm{I_{OP}} \rangle \rangle \tag{1}$$

where $\mathrm{N_{ST}}$—the name of the Web service, $\mathrm{D_{ST}}$—a text description of a Web service, $\mathrm{OPs_{ST}}$—set of Web service operations. Each of the Web service operations, in turn, is determined by $\mathrm{N_{OP}}$—name of the operation, $\mathrm{D_{OP}}$—text description of the operation, $\mathrm{O_{OP}}$ and $\mathrm{I_{OP}}$—outputs and inputs of operations.

Service template is compared with the set of Candidate Services (CS)—service templates obtained by analysis of Web services from a predefined set of Web services.

Comparison used to assess the syntactic and semantic similarity is:

$$\Theta(\mathrm{ST}, \mathrm{CS}) = \frac{W_H H(\mathrm{ST}, \mathrm{CS}) + W_\Phi \Phi(\mathrm{ST}, \mathrm{CS})}{W_H + W_\Phi} \tag{2}$$

where $\Theta(\mathrm{ST}, \mathrm{CS})$—the total assessment, $H(\mathrm{ST}, \mathrm{CS})$—syntactic similarity, $\Phi(\mathrm{ST}, \mathrm{CS})$—semantic (functional) similarity, w_i—the weight coefficient corresponding to each type of similarity designed for more flexible management of comparison criteria.

Syntactic similarity is computed similarly to basic method [1] using stemming, i.e., process of finding bases of words representing the name or description, and further calculation of the Hamming distance—the number of positions in which the symbols of the two words are different.

The algorithm used to find the base of word is based on a table of rules for converting an input word in the form of normalized one. Rules’ table defines prefixes and suffixes, which must be removed from the words or substituted by other prefixes or suffixes. This approach is preferable, because it shows good performance, while it is still simple enough. For example, an algorithm that uses a table to define rules of full word substitution is less efficient in common, while providing greater efficiency for languages with complex rules for the formation of prefixes or suffixes [8].

Semantic similarity is computed using a variety of criteria, which take into account the assessment of the similarity of two Web services by using the attributes and relationships of concepts.

The greatest impact on the evaluation of semantic similarity has its component—the property similarity of concept, which describes Web service operation (Fig. 1).

Calculation of property similarity stands for alternately comparing two pairs of properties selected for comparison concepts. This should provide the best possible

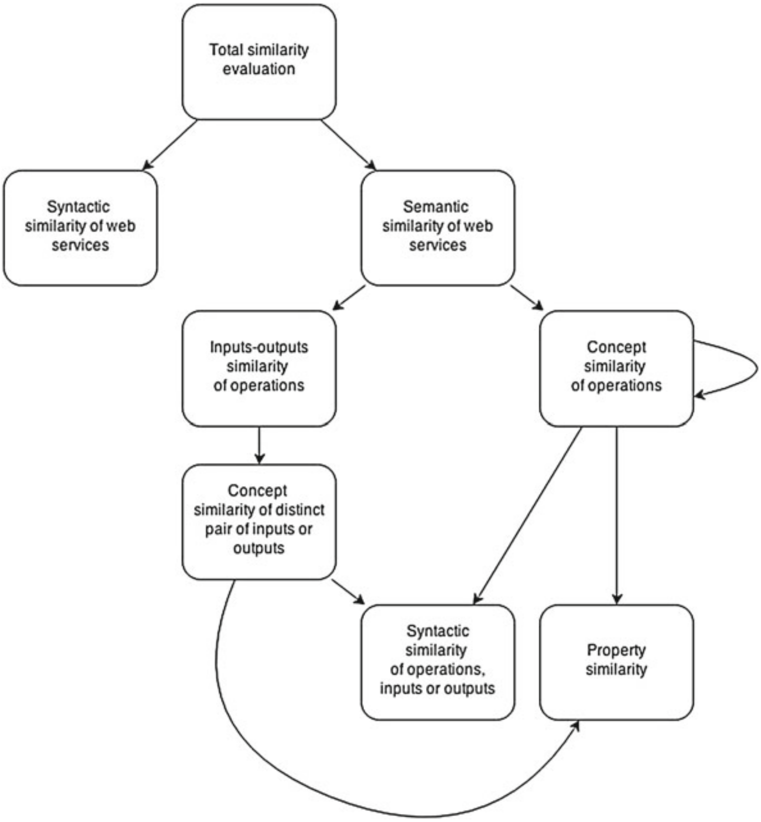

Total similarity evaluation
Syntactic similarity of web services
Semantic similarity of web services
Inputs-outputs similarity of operations
Concept similarity of operations
Concept similarity of distinct pair of inputs or outputs
Syntactic similarity of operations, inputs or outputs
Property similarity

Fig. 1 Method flow summary

estimate of the average similarity of properties:

$$P(C_{\mathrm{ST}}, C_{\mathrm{CS}}) = \max(P(CP_{\mathrm{ST}} - CPi_{\mathrm{ST}}, CP_{\mathrm{CS}} - CPi_{\mathrm{CS}} + p(CPi_{\mathrm{ST}}, CPi_{\mathrm{CS}})) \quad (3)$$

where $P(C_{\mathrm{ST}}, C_{\mathrm{CS}})$—property similarity assessment, CP_{ST} and CP_{CS}—set of properties describing the operation of service template and candidate-service respectively, CPi_{ST} and CPi_{CS}—distinct properties describing the operation of service template and candidate-service respectively. Similarity of distinct properties $p(CPi_{\mathrm{ST}}, CPi_{\mathrm{CS}}))$ is described as follows:

$$p = k \cdot \sqrt[3]{H \cdot K \cdot \chi} - 0.05 \cdot \|\text{properties with no reflection}\| \quad (4)$$

where *H*—syntactic similarity, *K*—similarity in cardinality, χ—similarity in constraints.

K is a constant factor equal to 1 (in case two properties are inverse functional) or 0.8–otherwise. Inverse functional are such properties that have values unique to each instance of an object. In other words, it defines its identity within the subject area.

Similarity in constraints is calculated by taking into account the limitations of the data type that represents a property. Property concept can be represented by a primitive type or another concept. Proceeding from this, there are three possible cases:

1. Both comparable properties are of primitive data type. Evaluation of similarity in constraints leads to calculating the amount of information lost by converting from one type to another. In practice, heuristic estimates are ranging from 0 (complete loss of information) to 1 (complete preservation of information).
2. Both properties are compared as ontological concepts. The task of evaluating the properties of the similarity of the two concepts is reduced to a recursive computation and evaluation, and it is the task of dynamic programming.
3. One of the properties is a primitive type, and another is a concept. In this case, the score is zero, because the types are incompatible. Such an assessment may be undesirable, because this case can arise from different levels of detail of compared concepts; some default estimates should be used in this case, or properties should be compared using syntactic similarity.

Similarity in cardinality also plays a significant role when comparing the two properties. Evaluation of similarity in cardinality involves evaluation of the cardinality of the set of values of the property taken. The cardinality of property may be finite value (in the case of an enumeration type), and conditionally finite value (for example, real numbers, which number of possible values is limited only by the specific implementation of environment).

Evaluation of similarity in cardinality is also empirical and is equal to:

1. 1, in case the cardinalities of two properties are equal;
2. 1, in case both properties are inverse functional properties;
3. 0.9, if the cardinality of property from ST is less;
4. 0.7, if the cardinality of property from ST is greater.

In case the number of service-candidate concept properties is not less than the number of service template concept properties, it is possible to make a "one-to-one" reflection of each template property. Otherwise, in case the reflection has the form "one-to-many," penalty should be applied proportional to the amount of properties that have multiple mappings. The heuristic value was set at 0.05.

Finally, after comparison each service-candidate receives the normalized in range [0, 1] assessment of similarity. The assessment of the value of 1 corresponds to the best fit in the sample. Then, Web services that meet the minimum allowable assessment are returned to the user in an ordered list.

4 Results

The method was tested on a set of Web services to assess response relevancy to a user request. Web services were obtained from the index of xIgnite and annotated with two different ontological models. The first was obtained by analyzing the domain of stock exchanges with materials of NASDAQ, which are publicly available on the Internet. The second ontology was created based on the concepts used by the company xIgnite to annotate own Web services.

First, consider four scenarios that can occur when comparing the two ontological concepts in single-ontological (within one ontological model), and four scenarios concerning multi-ontology environment (within a few ontological models). These are different possible kinds of situations for the comparison (in single-ontological "equivalent" concept means "identical"):

1. Equivalent concepts;
2. Concept and its sub-concept;
3. Concept and its super-concept;
4. Unrelated concepts.

Set used for the analysis of Web services includes 14 different Web services, 13 of which contain two operations and the remaining only one operation.

Simulation was performed in two stages. At the first stage service template that matches user's request, is described in advance defined (basic) concept of ontological model. Then, for each of the eight scenarios described above, candidate-service is described by one of the concepts that satisfy the conditions of the scenario; and then assessment is done.

The second step is to annotate operations of Web services from a set, and their properties, inputs, and outputs with previously selected base concept. Next, the assessment is done by using the base and extended methods.

Figure 2 shows the results of the first stage of the simulation. Scenarios 1–4 correspond to single-ontological environment and scenarios 5–8 correspond to multi-ontological one.

Our results imply that the total score using syntactic and semantic similarity estimates are only below the syntactic similarity for scenarios 3, 4, 7, and 8. Therefore, we consider that the semantic similarity reduces the total score for unrelated concepts and concepts that are unable to fully cover properties of the base concept. This in turn reduces the probability of the false matching.

Score of scenario 5 is also decreased, which is also associated with incomplete coverage properties of the super-concept, that there is a consequence of differences between the level of detail of ontological models.

Analyzing scenarios 1, 2, and 6, we can conclude that the total score improved for concepts satisfying basic properties of the concept (in this case—identical and sub-concepts of the basic concepts). Thus, when discovering a Web service, preference is given to Web services, which are included in this category.

Figure 3 shows the results of the second stage of the simulation. Web service operations 1–7 have more properties than the base concept can provide. Also the

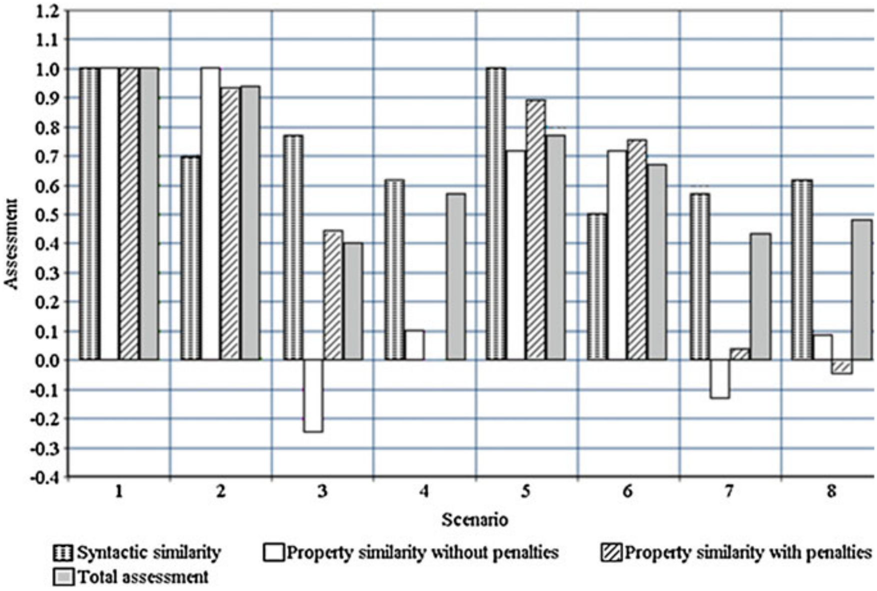


Fig. 2 Similarity assessment for different criteria

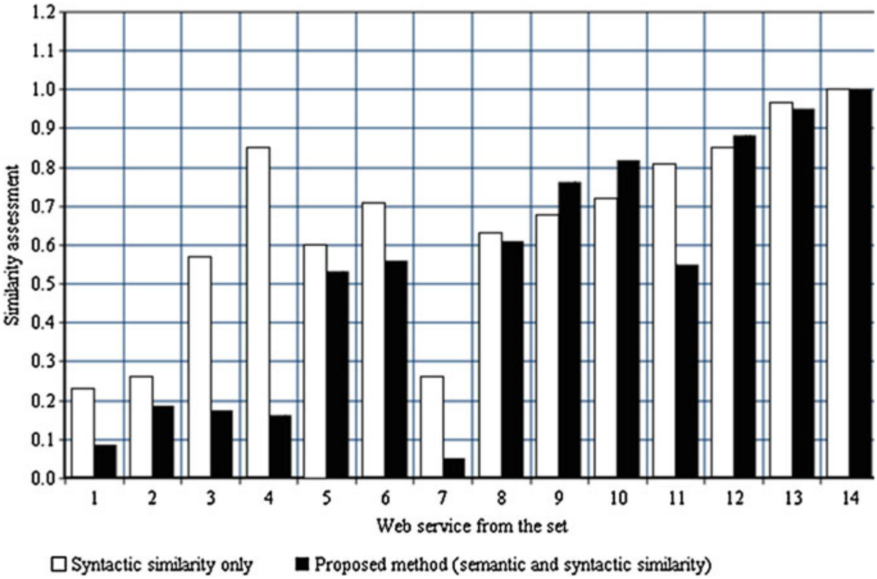


Fig. 3 Comparison of assessment received for different Web services

seventh one provides only one operation, while the service template describing a user request includes two operations.

According to the results, the method that takes into account the syntactic and semantic similarity has shown lower assessment than only syntactic similarity for Web services 1–7. Thus, the Web services which cannot fully cover the requirements of the service template are eliminated.

Given with the minimum accepted value of 0.7 for assess similarity, we find that using the proposed method, we obtain five Web services, and using the method that takes into account only the syntactic similarity we obtain eight of them.

Consequently, the semantic similarity assessment helps to reduce the probability of false matching (in this case—Web services 4, 6, and 11 are false matches).

5 Conclusion

The Web service discovery is completely a semantic problem. The current standards based on UDDI and WSDL do not use semantic data, therefore the relevancy of discovery response is still not high.

The solution for improvement of Web services discovery in multi-ontological environment method is proposed. Proxy abstract object called "service template" is proposed to be added and semantic similarity including additional criteria is introduced. It was shown that such extension can increase the relevancy of the Web service discovery response.

The further research is applied on extension of the proposed method by addition of support of the errors in terms of WSDL and requirements to QoS. Optimal values for evaluations have to be defined instead of heuristic ones by using of expert systems. Also the proposed method has to be verified on the larger set of Web services and their ontological descriptions.

The proposed method can be extended with addition of the composition feature, which will allow building a chain of web services operations called subsequently. It could be useful in case there is no target Web service found in result of discovery; however the chain exists, which can provide the same features as target Web service. It is proposed to perform such composition by using ontological descriptions of inputs and output of operations of each Web service.

References

1. Sheth, A.: Changing focus on interoperability in information systems: from system, syntax, structure to semantics. Interoperating Geographic Information Systems. Academic Publishers, Kluwer (1998)
2. Schmidt, C., Parashar, M.: A peer-to-peer approach to web service discovery. Internet and Web Information Systems. Kluwer Academic Publishers, World Wide Web (2003)
3. Klein, M., Bernstein, A.: Searching for services on the semantic web using process ontologies. The First Semantic Web Working Symposium (SWWS-1). Stanford, USA (2001)

4. Gonzales-Castillo J., Trastour D., Bartolini C.: Description logics for matchmaking of services. In: Proceedings of the Workshop on Application of Description Logics (2001)
5. Trastour, D., Bartolini, C., Gonzalez-Castillo, J.: A semantic web approach to service description for matchmaking of services. In: Proceedings of the 1st Semantic Web Working Symposium, CA (2001)
6. Sycara, K., Lu, J., Klusch, M., Widom, S.: Dynamic service matchmaking among agents in open information environments. J. ACM SIGMOD Rec. (1999) (Special Issue on Semantic Interoperability in Global Information Systems)
7. Magnini B., Serafini L., Speranza M.: Linguistic based matching of local ontologies. In: Working notes of Mean-02 (Workshop held in conjunction with AAAI-2002), Edmonton,July 28–August 1, 2002
8. Porter, M.F.: An algorithm for suffix stripping. Progr. Electron. library Inf. Syst. 14, July 1980

Advanced Approach to Web Service Composition

D. Pukhkaiev, O. Oleksenko, T. Kot, L. Globa and A. Schill

Abstract Web Service Composition (WSC) is a process that helps to save much programming and cost effort by reusing existing components—Web services. This process consists of two major stages—Web Service Discovery and Selection (WSD, WSS). This paper presents an overview of the current state-of-the-art WSD and WSS methods. It also provides an analysis and highlights major problems like lack of support of the syntactical description in fuzzy logic algorithms in WSD and complex approach shortage in WSS problem. Moreover, WSC approach and Service-level agreement (SLA) aware WSC System are presented.

Keywords Web service composition · Web service discovery · Web service selection · QoS · SLA

D. Pukhkaiev · O. Oleksenko · T. Kot (✉) · L. Globa
Information Telecommunication Networks Department,
National Technical University of Ukraine, Kyiv Polytechnic Institute,
Peremogy Ave. 37, Kyiv 03056, Ukraine
e-mail: tkot@its.kpi.ua

O. Oleksenko
e-mail: ooleksenko@stud.its.kpi.ua

D. Pukhkaiev
e-mail: dpukhkaiev@stud.its.kpi.ua

L. Globa
e-mail: lgloba@its.kpi.ua

A. Schill
Faculty of Computer Science, Dresden University of Technology, Nöthnitzer Str. 46,
01187 Dresden, Germany
e-mail: alexander.schill@tu-dresden.de

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_29

1 Introduction

Currently, many companies offer their services on the Internet. This creates a demand for tools to perform WSC, in particular WSD and WSS.

WSD may be defined as the process of finding a machine-processable specification of a Web service that meets certain functional criteria. In this paper, we offer a survey of Semantic Web service discovery approaches and define the main problems that should be solved in existing approaches in order to be used in real-world WSC system.

WSS is the next step in performing WSC. Overall goal of WSC is to provide end-user with fully working application, composite Web service, which satisfies his needs. Thus, an important aspect is to ensure that the composite Web service does not violate any nonfunctional properties, i.e., Quality of Service (QoS) parameters. Parameters such as response time, availability, robustness, reliability, and many others form user's experience and feedback indicating WSC efficiency.

However, despite much research effort, the state-of-the-art methods of Web service selection with QoS parameters taken into consideration cannot solve the problem of WSS in complex, focusing only on narrow tasks. Such tasks as improving speed of composition [1] or focusing on user preferences [2] are important aspects, but solving one task and neglecting others is a problem which needs to be solved. In this paper, an approach that overcomes the above-mentioned problems, as well as WSC System that performs QoS-aware Web service selection are presented.

The remainder of this paper is structured as follows. Existing approaches for WS discovery and their main shortcomings are discussed in Sect. 2. SLA-aware approach for WSS and WSC SLA-aware System are introduced in Sect. 3. Real-world scenario of composite Web service development using WSC System is shown in Sect. 4. Conclusion and future work are specified in Sect. 5.

2 Discovery

A WSD stage can be basically defined as a matchmaking process. Matchmaking is the process of finding an appropriate service provider for a service requester through a middle agent.

2.1 Definition of Comparative Evaluation Criteria

Growing number of Web services and ways to specify their functionality makes their discovery more and more difficult. A lot of algorithms and approaches were proposed to solve this discovery issue. In this chapter, criteria that allow us to evaluate and compare them are introduced.

Criteria were divided in three main groups—quantitative criteria, matching criteria, and technology support criteria.

Quantitative criteria are:

- Response time. To define how long it takes to process a WSD query.
- Performance. The WSD stage may be considered as a special Information Retrieval (IR) problem [3]. For IR systems evaluation, two following measures are used: recall and precision. Recall is a subset of the relevant documents that are retrieved. Precision is a fraction of retrieved by matchmaker results that are relevant. High performance indicates that discovery algorithm has both high recall and precision.
  Matching criteria are:
- Matching elements. The parts of WS specification that are used in matchmaking process. Possible options are:
  - IO: Inputs and outputs.
  - PE: Preconditions and effects or post-conditions.
  - Nonfunctional parameters.
- Multistage matching. To perform a discovery in several stages, sequentially or in parallel on different elements, followed by merging the results. This approach leads to more accurate results through increasing the matching complexity, which in turn increase the query response time. Thus, it is necessary to achieve a balance between accuracy and response time in such approaches. Some of them allow users to manage the trade-off between accuracy and response time [4].
  Technology support criteria are:
- Support for UDDI. Initially, all discovery approaches used UDDI syntax for matchmaking. However, while data in UDDI registries are stored using Extensible Markup Language (XML), semantic discovery approach can support UDDI only by combining the ontology matchmaking and UDDI semantic matchmaking.
- Support for different ontologies. Web services are autonomous, heterogeneous, and developed independently, using different ontologies in requester and provider sides. Support for different ontologies indicates that ontology conversion can be performed and Web services with different ontologies may be used [4].
- Support of probabilistic languages. In real-world systems, the common issue is incomplete information about the Web service functionality and user preferences for service discovery. A solution for this problem may be by using fuzzy, probability, and possibility theory [5]. Support for probabilistic extensions of semantic Web languages like pOWL, fuzzyOWL, or pDatalog is needed to compare semantic service annotations under uncertainty and with preferences.

2.2 Web Service Discovery Approaches

Discovery can be based both on the textual descriptions (Syntax-based discovery), and on the additional semantic descriptions (Semantic-based discovery).

Syntactic methods search through the text description of a Web service, keywords and qualifiers. Nonsemantic Web services can be discovered using UDDI [6]. UDDI is an industry specification for describing, publishing, and finding Web services.

Using UDDI developers can describe the functionality of their services and specify the technical details about the interaction with them. UDDI also defines a set of Application Programming Interfaces (APIs) that can be used for interaction with stored data.

The main advantage of syntax-based approaches is low response time due to simplicity of used algorithms (in comparison with semantic-based). They also do not require any other specifications except Web Services Description Language (WSDL).

The main disadvantage of such approaches is a necessity of manual selection from search results, which is eliminating the usage of this approach in fully automatic Web service composition systems.

Semantic Web service is a "web service which functionality is described by use of logic-based semantic annotation over a well-defined ontology" [5]. Due to the variety of semantic Web service description languages and means of service selection, different discovery approaches exist. The main approaches are:

- Logic-based Approaches. In this category of algorithms standard logic inferences are used. They determine the semantic relations between services on the basis of logical comparison of the service semantic descriptions. Strong mathematical basis makes logic-based approaches much more accurate than syntax-based approaches. Most of the semantic-based algorithms use this type of matching [4].
- Nonlogic-based Approaches. Using formal logic leads to considerable increase in complexity of the system that makes usage of this approach time consuming with high computational complexity. The nonlogic-based semantic Web service discovery aims to overcome such disadvantages. This category does not make semantic descriptions comparison of services and, instead, rely on such techniques as graph matching, information retrieval, and data mining.
- Logic- and Nonlogic-based Approaches. Usage of exclusively explicit semantics for similarity evaluation in logical approaches makes them inadequate. In such case, some relative services can be dropped from the answer set. To improve it, nonlogic-based approaches using both implicit semantics of services and logic approaches [4] may be applied. The basic idea of the Logic- and nonlogic-based approaches is that nonlogic-based matching techniques may be applied in case of a logic-based matching failure.
- Logic- and Syntax-based Approaches. Approaches from this category use both Logic-based matching and Syntax-based discovery.

2.3 Comparison

Results of the algorithm comparison are shown in Table 1, from which matchmaking categories may be compared with respect to two criteria: response time and performance.

For the comparison of algorithms by the performance criterion experiments provided by [7] are used. The performance has been evaluated based on the recall and

Table 1 Comparison of discovery approaches

Approach	Algorithm	Criteria								
		Quantitative criteria		Matching criteria				Technology support criteria		
		Response time	Performance	Matching elements			Multi stage matching	UDDI	DO	PL
				IO	PE	Nonfunctional				
Logic-based	OWL-S IDE (Srinivasan+ 06) [11]	Average	Average	+	+	–	–	–	–	–
	(Somasundaram+ 06) [12]			+	–	+	–	+	–	–
Nonlogic-based	(Li+ 07) [13]	Low	Low	+	–	–	+	+	+	–
logic- and nonlogic-based	SAWSDL-MX2 (Klusch+ 09) [9]	High	High	+	–	–	+	–	–	–
	FuzMOD (Ngan+ 07) [10]			+	+	–	–	–	+	+
Logic- and syntax-based	FUSION (Kourtesis+ 08) [8]	High	Average	+	–	+	+	–	–	–

precision measures. Based on these results it may be defined that integration of Logic-based and Nonlogic-based methods leads to inclusion of the syntactically similar, but logically disjoint results to the result set. This leads to higher performance of algorithms. Additionally, generally Logic-based matchmakers result in higher recall and precision compared to Nonlogic-based ones. Finally, in most cases, the Syntax-based matching only limits the searching domain [8] that makes no difference with the Logic-based matching in terms of the performance.

For comparing the response time, the results of the evaluation experiments in [9] were mainly used. The Logic- and Nonlogic approach provides the highest response time. Better result has the Logic-based approach and the Nonlogic approach provides a significant improvement in speed. Combined Logic- and Syntax-based algorithms have the lowest response time, as it allows performing preliminary selection on syntactic description. However, existing algorithms that use such approaches have the following disadvantages:

- Inability to match the parameter PE, and, as a result, reduced precision, which reduces performance.
- Lack of support of different semantic description standards (see Sect. 2.1).

The most promising of the considered algorithms is FuzMOD [10], since it is the only algorithm that supports incomplete information about the Web service functionality and user preferences for service discovery due to the usage of fuzzy logic in algorithm. However, it does not support the syntactical (textual) description discovery, which makes it impossible to work with one of the most common publishing Web services technology—UDDI.

One way for solving this issue is to use hybrid Logic- and Syntax-based algorithm. The main idea behind this algorithm is the separation of description processing—if semantic discovery mechanism fails, basic keyword search is being used, though results of such matching are considered less reliable. For semantic part of such algorithms, FuzMOD may be used. Keyword search may be based on Jaro–Winkler algorithm. It could be enhanced by synonym search using WordNet service. Development of such algorithm is a subject of future work.

3 Selection

3.1 Web Service Selection Description

Web Service Selection is a second step in WSC. It starts when the list of Web services with functional parameters is already created. The main goal of this stage is to select Web services with the best possible nonfunctional parameters, also called QoS parameters. Violation of these parameters such as performance, reliability, accessibility, availability, scalability, cost etc. can significantly affect the run of the application or even fail it entirely. Thus, it is very important to take into account QoS parameters and perform SLA-aware composition of Web services [14].

3.2 Comparison of the State-of-the-Art SLA-aware WSS Approaches

Various researches have been conducted to investigate the subject of SLA-aware or QoS-aware WSS. These approaches have different goals and view WSS from different perspectives.

The preference-based approach [2] calculates composite service's QoS taking into account price, response time, reliability and reputation. Moreover, it uses coefficients based on user preferences to prioritize or another requirements.

Heuristic approach [15] divides QoS parameters into three groups: additive parameters, multiplicative parameters, and attributes aggregated by Min-operator. Also this approach provides SLA monitoring and reconfiguration.

Genetic algorithm [1] uses decomposition of global QoS constraints of composite Web service into local ones for every Web service. Then, it uses linear search to choose the best simple Web service. Two groups of QoS parameters are used: positive (availability and throughput) which are maximized and negative (price and response time) which are minimized. Good performance during runtime is the main focus of this approach. Possibility of monitoring SLA is stated, but no mechanisms are presented.

The Breadth First Use algorithm [16] utilizes only response time and throughput. This implies the low quality of the composition. Moreover, monitoring phase is not introduced.

Analysis of these approaches shows that only heuristic and genetic approaches cover sufficient number of QoS parameters. However, they do not support subjective QoS parameters which are necessary to compose optimal composition from user's perspective. Monitoring phase support is also a bottleneck while only heuristic approach is able to perform it.

This comparison has shown that the most reliable is heuristic approach. However, it lacks flexibility, especially in areas of new user-defined QoS parameters, objective QoS parameters support and user preferences. Table 2 summarizes the results of comparison.

Thus, development of SLA-aware WSS approach which is able to stand up to all the requirements provided in this section is an important task.

Another important issue is to unite the approach of WSS with WSD—such combination provides significant step comparing to state-of-the-art methods described above.

Table 2 SLA-aware WSC approaches comparison

	Preference-based	Heuristic	Genetic	Breadth first use
Full stack of QoS parameters	−	+	+	−
Subjective QoS	+	−	−	−
Monitoring	−	+	±	−

3.3 SLA-aware WSS Method and WSC System

In this subsection, general description of SLA-aware WSS method and corresponding software implementation is provided. WSC is a broader concept than WSS. The SLA-aware WSC System should be able to perform both tasks of WSD and WSS. More detailed description is given in [14], although the main focus is on WSS.

Basic approach consists of seven steps:

- Extracting of discovery parameters from the workflow design;
- Matchmaking with providers Web service specification;
- Generating list of matching Web services;
- Extracting input parameters—list of Web services which satisfy functional parameters from Web service discovery service;
- Utilization of integral indicator of Web service quality compliance in order to grade found Web services by nonfunctional parameters;
- Web service selection and composition itself;
- Runtime monitoring and reconfiguration.

Suggested WSC System which comprises WSD and WSS consists of five major blocks: service locator, SLA extractor, decision maker, service combiner and service monitor.

Service locator block is intended to find Web services satisfying functional parameters provided by workflow design stage—Business Process Model and Notation (BPMN) file. This corresponds to the discovery stage of WSC. Found Web services are organized into a list, sorted by the integral indicator of Web service quality compliance for each activity.

SLA extractor block extracts QoS information from WS-Agreements and provides decision maker module with nonfunctional parameters values. Decision maker calculates rankings due to ontology rules considering user preferences provided by the client. These preferences have higher priority than ontology rules. Thus, QoS parameters of composite Web service fulfill subjective QoS parameters support constraint. Service combiner combines selected services into executive Business Process Execution Language (BPEL) file.

Service monitor identifies changes of QoS parameters and reconfigures composite Web service in case of their violations.

Integral indicator of Web service quality compliance is the key parameter in composite Web service evaluation. It shows the ranking of a Web service for possible composition options. Thus, WSC System can choose the best composite Web service regarding QoS parameters. Comparing to the QoS of a single Web service (which is a part of a composite Web service), ranking of a composite one is not a trivial task. QoS parameters for a composite service depend on the initial workflow. Calculations of QoS parameters for a composite service are presented in [14].

Applying of user-defined rankings changes the priority of QoS parameters for composition. Thus, client receives a service which satisfies his needs. If the client decides not to specify any priority, default values of rankings will be applied.

Integral indicator of Web service quality compliance can be presented as:

$$\mathrm{Nf} = \mathrm{Operator}(\mathrm{R_i Q_o Sp_i}) \tag{1}$$

where $\mathrm{QoSp_i}$—one of the QoS parameters (e.g., performance, reliability, robusteness, accessibility etc.), R_i—ranking of corresponding QoS parameter. $\mathrm{QoSp_i}$ has a value from 0 to 1 proportionally to the actual value of the parameter. Operator in context of formula (1) can be overridden by the sum, multiplication, max or power operator depending on the workflow. In particular, the operator depends on composition pattern of Web services (loop, sequence etc.) and QoS parameter itself [14].

Several workflow and WSC models which provide realization of proposed approach have been developed. Workflow model on design stage is presented in [17]. Workflow model on enactment stage and WSC model are given in [14].

4 WSC System

This section provides presentation of WSC System and WSC approach based on possible real-world scenario.

4.1 Architecture Overview

Figure 1 depicts the Business Concept Model (BCM) according to [18] approach. This model represents the very basic view on the WSC System. It does not contain any implementation specific information.
The Business Concept Model consists of following concepts.

- User. The client of the system. He is provides an abstract BPEL file to the system.
- Abstract BPEL. A file which is provided by the user of the system, it doesn't contain any specific Web services links, but only the information necessary for the Service Discovery stage.

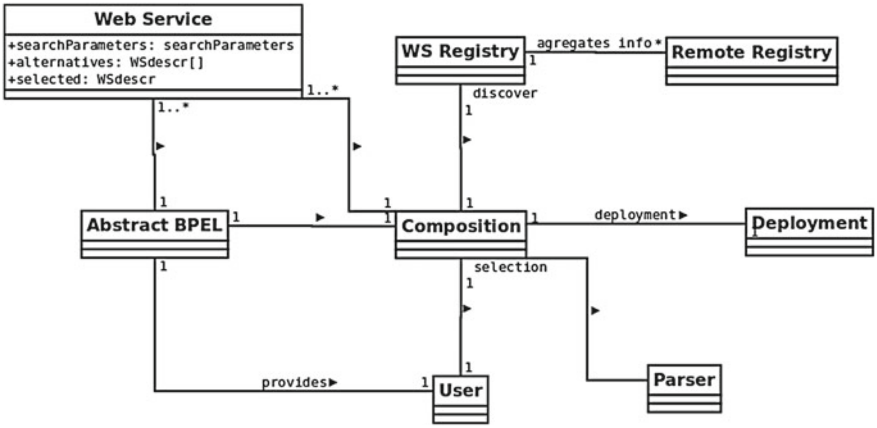


Fig. 1 Business Concept Model

- Composition. The core element of the BCM. It glues other concepts thus being able to provide Service Discovery, Selection and Deployment.
- Web Service. A component of the Composition which is represented by a composite Web service.
- WS Registry. Aggregates and provides a specific view on Remote Registries' Web Services. The core concept of the WSD stage.
- Remote Registry. UDDI or another registry which contains the list of Web services and descriptions.
- Parser. Based on the Service Selection stage parses and generates a concrete BPEL file out of the abstract. Modifies selected services WSDL files in order to use them in composition.
- Deployment. A concept which comprises functions necessary to deploy the composite Web service.

Figure 2 depicts the Business Interface Model (BIM) [18]. This model provides more detailed view on the WSC System. Here only components related to the actual WSC System remained. Core business types of the system are identified in the BIM. This means that these entities can exist independently of the other components' existence. Thus they can be interchanged with other implementations which are able to provide the same functionality. All core types have business interfaces in order not to expose internal structure therefore preserving encapsulation. Finally, the BIM provides key fields of each business type which are required to provide the declared functionality. Figure 3 depicts the Initial Component Specification Architecture (ICSA) [18]. It is defined based on the interfaces of the BIM, one component specification per interface. Since management interfaces were created to manage instances of core business types and their associated details, they are concerned with information that is managed independently. It leads to separate component specifications for Parser,

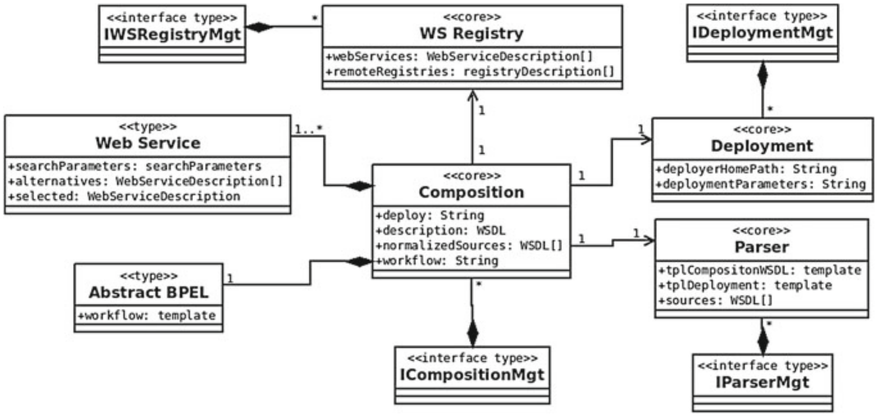


Fig. 2 Business Interface Model

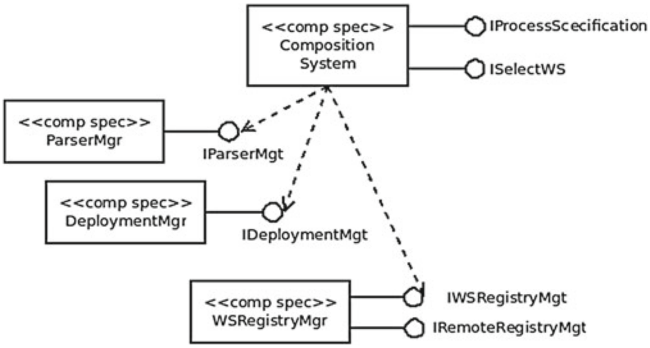


Fig. 3 Initial Component Specification Architecture

Deployment, Web Service Registry, and Composition types. After that these separate components can be bound together into the ICSA.

4.2 Case Study

Assume that the client of WSC System has a goal to develop service, providing vacation. Customization in this context means that the end-user would be able to book a hotel and flight, taxi, and tickets to some entertainment events using just one service. Lack of funds, programming skills, or time implies into using third-party services.

Client has various requirements to his service, e.g., response time, cost etc. After authentication in WSC System, client can start developing his service. BPMN or BPEL file has to be uploaded in order to provide system with workflow information.

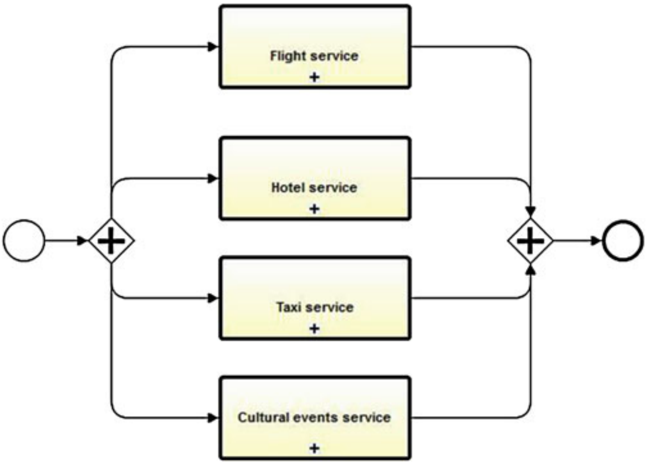


Fig. 4 Simplified BPMN diagram of provider's application

Next step is to specify QoS requirements. If none were provided, system will eventually find the best possible solution. However, even the most reliable one may not satisfy users' expectations. Thus, it is strongly recommended to provide application with nonfunctional parameters values. When QoS parameters are specified, WS-Agreement for the composite service is generated. In Fig. 4 BPMN diagram for Vacation Service is presented. The exact workflows are omitted for simplicity.
The Service locator extracts the information about functional parameters from BPEL file which was either uploaded or generated from BPMN. It also searches the appropriate services in UDDI or service brokers (considering functional parameters).

Client receives list of composite Web services (combination of simple Web services) satisfying functional parameters sorted by the integral indicator of Web service quality compliance.

In system settings, the client can choose whether composition will be done automatically or ask for human interaction. Eventually the client has to choose the composition that he prefers from the list and confirm the purchase of corresponding Web services. After this, the client has a functioning composite Web service.

In the runtime, service monitor identifies changes of QoS parameters and reconfigures service in the way similar to initial WSC described above or asks for human interaction.

In case of violating functional parameters, composite Web service is recomposed from scratch. Such state of the service cannot be allowed, because service does not provide declared functions. Eventually, the end-user works with Web interface where all single services are combined transparently.

5 Conclusion

Web Service Composition consists of two major blocks: Web Service Discovery, finding Web services satisfying functional parameters and Web Service Selection, choosing the best possible combination of Web services regarding functional parameters.

Despite much research effort still many problems exist. WSD problem is lack of the syntactical description support in fuzzy logic algorithms. WSS problem is a narrow task focusing and thus neglecting of other important aspects.

Presented SLA-aware WSC System is able to solve the problems of Web service Discovery as well as Web service Selection. The SLA-aware WSC system covers such aspects as full stack of QoS parameters support, subjective QoS, i.e., user preferences and monitoring stage support.

Another important issue is synchronous utilization of WSD and WSS. It means that the presented approaches are fully compatible. Thus, SLA-aware WSC System is able to perform full WSC.

Future work is aimed on integration of WSD fuzzy logic approach with the support of syntactical description into the overall system. After the integration, comprehensive system testing will be applied and quantitative results are provided. Also, the tutorial for WSC System is to be written.

References

1. Mardukhi, F., NematBakhsh, N., Zamanifar, K., Barati, A.: QoS decomposition for service composition using genetic algorithm. Appl. Soft Comput. **13**(7), 3409–3421 (2013)
2. Wei, Z., Junhao, W., Min, G., Junwei, L.: A QoS preference-based algorithm for service composition in service-oriented network. Optik—Int. J. Light Electron. Opt. **124**(20), 4439–4444 (2013)
3. Küster, U., Lausen, H., König-Ries, B.: Evaluation of semantic service discovery—a survey and directions for future research. In: Emerging Web Services Technology, vol. 2, pp. 41–58 (2008)
4. Keyvan, M., Suhaimi, I., Mojtaba, K., Kanmani, M., Sayed, G.H.T.: A comparative evaluation of semantic web service discovery approaches. In: Proceedings of the IIWAS2010, Paris, 08–10 November 2010
5. Klusch, M.: Semantic web service coordination. In: CASCOM: Intelligent Service Coordination in the Semantic Web, pp. 59–104 (2008)
6. Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S.: Unraveling the web services web: an introduction to SOAP, WSDL, and UDDI. IEEE Internet Comput. **6**, 86–93 (2002)
7. Klusch, M., Fries, B., Sycara, K.: OWLS-MX: a hybrid semantic web service matchmaker for OWL-S services. Web Semant.: Sci. Serv. Agents World Wide Web **7**(2), 121–133 (2009)
8. Kourtesis, D., Paraskakis, I.: Combining SAWSDL, OWLDL and UDDI for semantically enhanced web service discovery. In: The Semantic Web: Research and Applications, pp. 614–628 (2008)
9. Klusch, M., Kapahnke, P., Zinnikus, I.: SAWSDL-MX2: a machine-learning approach for integrating semantic web service matchmaking variants. In: IEEE International Conference on Web Services (2009)

10. Le, D.N., Goh, A.E.S.: FuzMOD: a fuzzy multi-ontology web service discovery system. In: The 2nd IEEE APSCC, pp. 197–203 (2007)
11. Srinivasan, N., Paolucci, M., Sycara, K.: Semantic web service discovery in the OWL-S IDE. In: Proceedings of HICSS'06, System Sciences (2006)
12. Somasundaram, T.S., et al. Semantic description and discovery of grid services using WSDL-S and QoS based matchmaking algorithm. In: ADCOM (2006)
13. Li, H., Du, X., Tian, X.: A WSMO-based semantic web services discovery framework in heterogeneous ontologies IIWAS2010. In: Proceedings Web Services Environment. Lecture Notes in Computer Science, vol. 4798, p. 617 (2007)
14. Pukhkaiev, D., Kot, T., Globa, L., Schill, A.: A novel SLA-aware approach for web service composition. In: IEEE EUROCON, pp. 327–334 (2013)
15. Berbner, R., Spahn, M., Repp, N., Heckmann, O., Steinmetz, R.: Heuristics for QoS-aware web service composition. In: Proceedings of the IEEE International Conference on Web Services 2006, pp. 72–82. IEEE Computer Society Washington (2006)
16. Aiello, M., El Khoury, E., Lazovik, A., Ratelband, P.: Optimal QoS-Aware web service composition. In: IEEE Conference on Commerce and Enterprise Computing, pp. 491–494 (2009)
17. Kot, T., Reverchuk, A., Globa, L., Schill, A.: A novel approach to increase efficiency of OSS/BSS workflow planning and design. In: Proceedings of BIS'12, vol. 117, pp. 142–152. Springer, Berlin (2012)
18. Cheesman, J., Daniels, J.: UML Components: A Simple Process for Specifying Component-Based Software. Addison Wesley, Boston (2004)

Based on Force-Directed Algorithms Method for Metagraph Visualization

Larysa Globa, Maksym Ternovoy, Olena Shtogrina and Oleksandra Kryvenko

Abstract This paper describes the method for automatic metagraph visualization based on the principles of force-directed algorithms. The criteria under which the final image is understandable for users and corresponds to a predetermined metagraph are defined. This approach defines the set of the rules for forces between metagraph nodes depending on the types of the nodes between which the forces act. The analogue of Venn diagram is used to visualize the metagraph nodes. The method was tested on random metagraphs with up to 60 vertices and up to 25 metavertices.

Keywords Metagraph · Visualization · Layout · Metavertex · Graphical representation · Graph

1 Introduction

Nowadays information visualization tools are actively developing. These tools are converting the data into a form that allows using an ability to analyze visual images. The methods of graphical analysis can increase efficiency in decision support process. To visualize the interrelated data it is better to use graphs. But there are a lot of cases when common graph theory cannot be used to visualize the data correctly. For example, we cannot describe and visualize vertices, nested vertices, and relation with them in graph terms. In these cases, the metagraphs can be used.

L. Globa (✉) · M. Ternovoy · O. Shtogrina · O. Kryvenko
Institute of Telecommunication Systems, National Technical University of Ukraine,
Kyiv Polytechnic Institute, Kyiv, Ukraine
e-mail: lgloba@its.kpi.ua

M. Ternovoy
e-mail: ternovoy@its.kpi.ua

O. Shtogrina
e-mail: L_Shtogrina@mail.ru

O. Kryvenko
e-mail: okryvenko@stud.its.kpi.ua

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_30

Metagraph is a graphical construct specified by its generating set and the set of edges defined on the generating set [4]. It can be used to model rule bases, data bases, model bases, business processes, etc.

There are many visualization algorithms for small (up to 200 nodes) and large (thousands of nodes) graphs. Visualization algorithms for small size graphs based on the physical analogues or force-directed algorithms are the simplest. They can be used to draw graphs of any kind. Images created with these algorithms meet the requirements of esthetics criteria: they contain few intersections of edges and symmetry [3]. This group includes force algorithms [6, 8], algorithms that use the action of gravity [7], magnetic forces [14], algorithms based on the minimization of energy [12] and others. Visualization of large graphs is a more complex problem. To solve this problem, other approaches are needed. Large graphs visualization requires algorithms that use graph clusterization, graph GC-filtration [10], forces approximation [3], multiscale methods [9, 10, 15], topological feature-based method [1], etc. These approaches are effective for graphs, but are unsuitable for metagraph because of different structure. Unlike graph visualization, there are no well-known algorithms for metagraph visualization. In this paper, we propose the method for automatic metagraph visualization.

2 State of the Art and Background

As mentioned above, metagraph is specified by its generating set and the set of edges defined on the generating set [4]. In [2], by analogy with the hypergraph theory, two components such as metavertices set and metaedges set are added to metagraph description. To solve the problem of metagraph visualization, we decide to extend the metagraph definition given in [4] through the vertices set and metavertices set consideration separately. The set of edges contains all metagraph edges, no matter what types of nodes they connect. Suggested metagraph definition is described below.

Definition 1 Metagraph is a construct $S = \langle V, M, E \rangle$, where
$V = \left\{ v_r | r = \overline{1, N_V} \right\}$—set of metagraph vertices,
N_V—number of metagraph vertices;
$M = \left\{ m_q | q = \overline{1, N_M} \right\}$—set of metavertices, N_M—number of metavertices;
$E = \left\{ e_h | h = \overline{1, N_E} \right\}$—set of edges, N_E—number of metagraph edges.

Metavertex $m_q = \left\{ v_r | v_r \in V,\ r = \overline{1, N_{m_q}} \right\}$ is a vertex which includes some other vertices which are called inner vertices for this metavertex, N_{m_q} is a number of vertices included in m_q.

Metagraph node $mv \in (V \cup M)$ is a vertex or a metavertex.

Metagraph edge is a oriented pair of vertices $e_h = (mv_{out}, mv_{in})$, where mv_{out}—tail, mv_{in}—head.

Graph visualization algorithms do not have the mechanism that allows including vertices in other vertices. If we apply these algorithms to metagraph there are layout problems. Suppose given metagraph S_1 (Fig. 1):

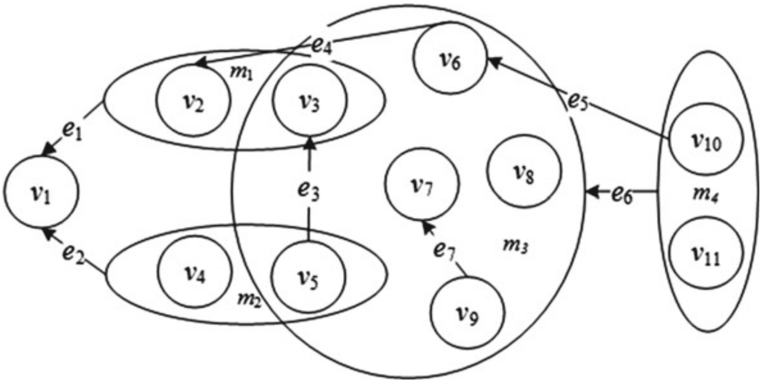


Fig. 1 Correctly positioned metagraph S$_1$

$$S_1 = \left\langle \begin{array}{l} \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}\}, \\ \{m_1, m_2, m_3, m_4\}, \\ \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\} \end{array} \right\rangle,$$

where $m_1 = \{v_2, v_3\}$, $m_2 = \{v_4, v_5\}$, $m_3 = \{v_3, v_5, v_6, v_7, v_8, v_9\}$, $m_4 = \{v_{10}, v_{11}\}$, $e_1 = (m_1, v_1)$, $e_2 = (m_2, v_1)$, $e_3 = (v_5, v_3)$, $e_4 = (v_6, v_2)$, $e_5 = (v_{10}, v_6)$, $e_6 = (m_4, m_3)$, $e_7 = (v_9, v_7)$.

Have a look at the ways to use graph visualization algorithm for metagraph. The first way is to interpret all metagraph nodes as graph vertices. In this case, the metavertices positions and their inner vertices positions are computed independently. Therefore inner vertices of metavertex may be positioned at a significant distance from each other. This leads to a loss of the metavertex form.

The second way is to take into account only vertices and then lead round inner vertices to visualize metavertex. In this case it is not guaranteed that only inner vertices are located in metavertex. Also incident to metavertex edges cannot be determined. Figure 2 shows the wrong location metagraph nodes when applied to the algorithm for graph visualization. Wrong located nodes are marked gray. Also there are no edges between metavertices.

3 Metagraph Layout Criteria

Some basic requirements for graphical representation of the graph are described in [3, 6–8, 14]. They are the minimum number of edge crossings, approximately equal edges length, the display of symmetries existing in the graph, etc. These requirements are also valid for metagraph. But the correct location of metavertices and their shape are more important for metagraph drawing. The proposed layout criteria are described below.

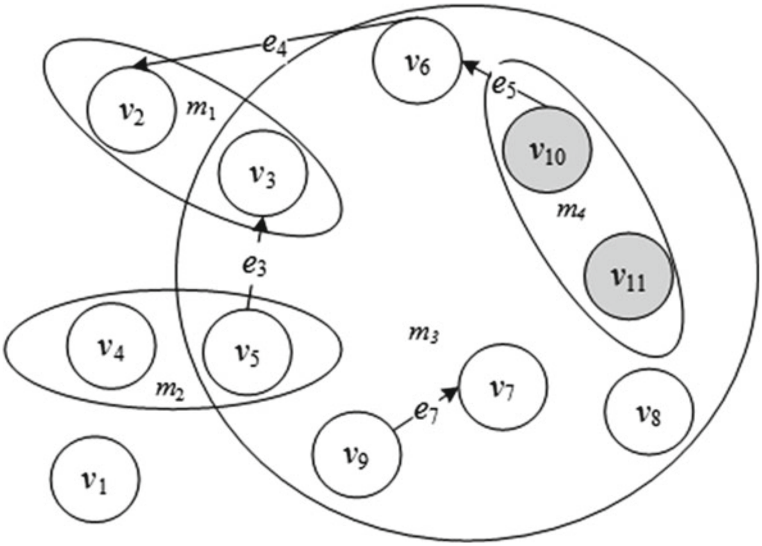


Fig. 2 Incorrectly positioned metagraph S_1

Each vertex v_r and metavertex m_q has a position $p_{v_r} = \left(x_{v_r}, y_{v_r}\right)$, $p_{m_q} = \left(x_{m_q}, y_{m_q}\right)$. $P_V = \left(p_{v_1}, p_{v_2}, \ldots p_{v_{N_V}}\right)$ is the vector of vertices positions, $P_M = \left(p_{m_1}, p_{m_2}, \ldots p_{m_{N_M}}\right)$ is the vector of metavertices positions. These vectors fully determine the locations of nodes for visualization. The distance between nodes is calculated as distance between points: $l_{e_h} = \left\| p_{mv_i} - p_{mv_j} \right\|$, where p_{mv_i}, p_{mv_j} are the designation of the ith and jth nodes coordinates correspondently.

Definition 2 Graphical representation of the metavertex is the geometric figure F_{m_q}, with position p_{m_q} that corresponds to the center of the figure.

Definition 3 Graphical representation of the edge is the curve F_{e_h} defined between the coordinates of the head and tail node.

Definition 4 Graphical representation of the metagraph is the set $W_S = \langle P_V, F_M, F_E \rangle$, where $F_M = \left\{F_{m_q}\right\}$ is the set of metavertex figures, $F_E = \left\{F_{e_h}\right\}$ is the set of edges curves.

We consider graphical representation is correct, if mapping $S \rightarrow W_S$ is isomorphic. We propose metagraph layout criteria, which consists of three requirements:

1. Coordinates of vertices are not equal: $\forall i, \forall j : i \neq j \Rightarrow p_{v_i} \neq p_{v_j}$. Metavertices coordinates can be equal in the presence of common inner vertices.

2. The only inner vertices coordinates are located in metavertex figure F_{m_q} Metavertex figure F_{m_q} contains the only inner vertices of the metavertex m_q:

$$\forall v_r{:}v_r \in m_q \Rightarrow p_{v_r} \in F_{m_q}, \quad \forall v_r{:}v_r \notin m_q \Rightarrow p_{v_r} \notin F_{m_q}.$$

3. Metavertices figures without the common vertices do not intersect: $\forall j, \forall k{:}m_j \cap m_k = \emptyset \Rightarrow F_{m_j} \cap F_{m_k} = \emptyset$.

4 The Method for Visualization

Problem statement for metagraph visualization.
Given:

1. Metagraph $S = \langle V, M, E \rangle$.
2. P_V, P_M—start location of vertices and metavertices.
3. Metagraph layout criteria.
4. Rectangular region U. The result image should be placed in U.

Find: W_S.
To solve this problem it is necessary to determine the placement of nodes that belong to metavertices and do not belong to them, location of metavertices that contain common vertices, relative position of vertices in metavertex figure, metavertex figures and curves F_{e_h}.

The proposed method is based on the Fruchterman and Reingold visualization algorithm [8]. Metagraph is represented as a system of objects, connected by springs according to certain rules. Each spring acts on the pair of nodes with the force of attraction or repulsion. Vertices move under the influence of the sum of these forces. The algorithm stops when the system reaches a point of equilibrium. The nodes are placed inside the rectangular area U.

With each iteration nodes move at a distance $\delta(t)$ in the direction of the total force acting on the node. Function $\delta(t)$ depends on the temperature and decreases. Repulsive force acts between each pair of nodes, except metavertex and its internal vertex.

The attraction force acts between:

- adjacent nodes;
- metavertices and their inner vertices (this force provides the location of inner vertices in metavertex shape and vertices movement for the metavertex);
- all vertices in metavertex (this force provides a smaller area of the metavertex figure).

Then, the repulsive force induced by mv_i and acting on mv_j is defined as:

$$f_{rep}\left(i, j\right) = Kr_{ij} \frac{l^2}{\left\| p_{mv_i} - p_{mv_j} \right\|} p_{0_{ji}}, \tag{1}$$

if $i = j$, $f_{rep}\left(i, j\right) = 0$.

Vector $p_{0_{ji}}$ is the direction vector from p_{mv_j} to p_{mv_i}. The optimal length l for an edge of metagraph is calculated as a function of allocation area and number of nodes:

$$l = \sqrt{\frac{area(U)}{N_V + N_M}}. \tag{2}$$

The attraction force induced by mv_i and acting on mv_j is defined as:

$$f_{attr}\left(i, j\right) = Ka_{ij} \frac{\left\| p_{mv_i} - p_{mv_j} \right\|^2}{l} p_{0_{ij}}, \tag{3}$$

if $i = j$,$f_{attr_{ij}} = 0$.

Also the force of gravity attracts every node to the center of metagraph similar to the algorithm [7].

$$F_{gr}\left(i\right) = \left(1 + \frac{\deg\left(mv_i\right)}{2}\right) \cdot K_{grav} \frac{p_{mv_i} - B}{\left\| p_{mv_i} - B \right\|}, \tag{4}$$

where $B = \frac{1}{N_V + N_M} \cdot \sum p_{mv}$—metagraph center,

$\deg\left(mv_i\right)$—node degree or the number of edges connected to it,

$\left(p_{mv_i} - B\right)$—the direction to the center of the metagraph.

Force of gravity keeps weakly connected nodes closer to whole metagraph. Its value depends on the node degree, so the nodes with many incoming or outgoing edges are located closer to the center. The presence of the force of gravity makes the overall arrangement more circular.

Then the total force acting on the node mv_i is calculated by the following formula:

$$F_{spring}\left(i\right) = \sum_j f_{rep}\left(i, j\right) + \sum_j f_{attr}\left(i, j\right) + F_{gr}\left(i\right).$$

The heuristics coefficients. The force values depend on the coefficients Kr_{ij} and Ka_{ij}. The coefficients depend on the type of each node and their relationship in a pair $mv_i$and mv_j. They can be presented in two matrices: the attraction coefficient matrix and the repulsion coefficient matrix.

$$|Kr| = \begin{pmatrix} 0 & Kr_{12} & \cdots & Kr_{1n} \\ Kr_{21} & 0 & & \\ \vdots & & \ddots & \\ Kr_{n1} & \cdots & Kr_{nn-1} & 0 \end{pmatrix}, \quad |Ka| = \begin{pmatrix} 0 & Ka_{12} & \cdots & Ka_{1n} \\ Ka_{21} & 0 & & \\ \vdots & & \ddots & \\ Ka_{n1} & \cdots & Ka_{nn-1} & 0 \end{pmatrix}.$$

Rows and columns of both matrices correspond to metagraph nodes. There are no repulsion force between pairs metavertex and inner vertex, so $Kr_{uv} = 0$.

Pairs $mv_i, mv_j \in m_q$ (inner vertex and inner vertex) have coefficient of repulsion:

$$Kr = Avg_{inner}.$$

Avg_{inner} is the average number of inner vertices in metavertices. The distances between inner vertices in metavertex should be approximately equal.

Other pairs have repulsion coefficient:

$$Kr_{ij} = \frac{W_{mv_i} + W_{mv_j}}{2},$$

where W_{mv_i} is the weight coefficient

$$W_{mv_i} = \begin{bmatrix} 1, & mv_i \in V \\ N_{m_q}, & mv_i = m_q \in M \end{bmatrix}.$$

Thus, if some vertices mv_i and mv_j do not belong to any metavertex, then $Kr_{ij} = 1$. If mv_i or mv_j is the metavertex, repulsion force depends on its metavertex weight coefficient. Metavertex weight coefficient is a number of vertices included in it. If mv_i and mv_j are metavertices, the repulsion force depends on the average value of weight coefficients. The distance between metavertices with big number of inner vertices is larger.

If $mv_j \in M$ is the metavertex m_q and $mv_i \in m_q$ then attraction coefficient is defined by the formula:

$$Ka_{ij} = \frac{N_{m_q} + \deg\left(m_q\right) + 1}{1 + \frac{\deg(mv_i)}{2}}.$$

If $mv_i, mv_j \in m_q$ is the inner vertices for metavertex m_q:

$$Ka_{ij} = \frac{Avg_{inner}}{1 + \frac{\deg(mv_i)}{2}}.$$

If some other nodes mv_i and mv_j are connected by an edge:

$$Ka_{ij} = \frac{W_{mv_i} + W_{mv_j}}{2 + \deg\left(mv_i\right).}$$

If there is no edge between other nodes mv_i and mv_j: $Ka_{ij} = 0$.

Nodes with high degree move slower because of the expression $1 + \deg(mv_i)/2$ is in the denominator.

Coefficient matrices are normalized relative to the maximum value. Therefore, elements in matrices less than or equal to one.

The algorithm of the method realization. The parameter ε is the accuracy of finding the equilibrium point; *stop*—the boolean parameter for determining the end of the algorithm, it is set as *true* in first iteration.

The proposed method for metagraph visualization consists of the following steps:

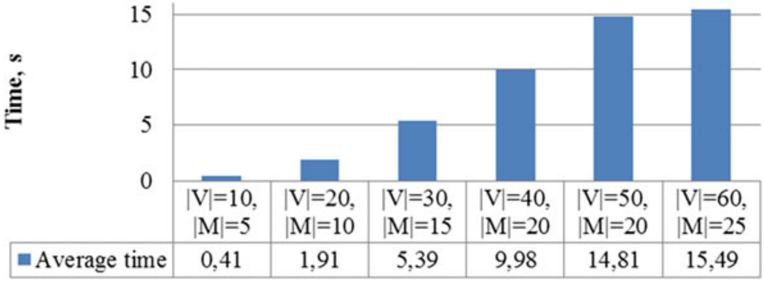
step:1 Calculate the optimal edge length using the formula (2).
step:2 Set the temperature t as maximum temperature.
step:3 Calculate repulsion matrix and attraction matrix.
step:4 Do steps 4.1–4.2 for each pair of nodes mv_i and mv_j
step:4.1 Calculate the sum of attraction forces (1), repulsion forces (3) and gravity force (4).
step:4.2 If $\left\| F_{spring}(u) \right\| > \varepsilon$ and is not reached the maximum iterations number, $stop = false$.
step:5 If $stop = true$ go to step 9.
step:6 Do steps 6.1–6.2 for each node.
step:6.1 Calculate new node position $p_{mv} = p_{mv} + \delta(t) \times \frac{F_{spring}(u)}{\left\| F_{spring}(u) \right\|}$.
step:6.2 Prevent p_{mv} from being displaced outside frame U.
step:7 Reduce the temperature.
step:8 $stop = true$, go to step 4.
step:9 Find F_{m_q} for each metavertex and set the metavertex position as the center of F_{m_q}.
step:10 Find F_{e_h} for each edge.

In the suggested method, the minimum convex hull was chosen as an optimal metavertices figure. To find convex hull Jarvis, Graham, and Chan algorithms [5] can be used. The edges can be set as a straight line connecting node shapes. In this case, there are multiple intersections of edges and intersections metavertex figures. Therefore, it is necessary to calculate the shape of the edges as Bézier curve or B-spline [11, 13].

5 Experiment and Results

The proposed method was tested on random metagraphs with up to 60 vertices and up to 25 metavertices and each metavertex had less than five intersections for any of its vertex. The tests were made on PC with Intel(R) Core(TM) i3 CPU 2 × 2.4 GHz, 3 Gb RAM.

It was observed that depending on the initial location of the nodes, the result image was different. The time required to obtain a satisfactory image depends on the



	|V|=10, |M|=5	|V|=20, |M|=10	|V|=30, |M|=15	|V|=40, |M|=20	|V|=50, |M|=20	|V|=60, |M|=25
■ Average time	0,41	1,91	5,39	9,98	14,81	15,49

Fig. 3 Average time needed for random metagraph visualization

metavertices number and the number of inner vertices. The time is reduced with the improvement of the initial location of the nodes. Figure 3 illustrates the average time needed for random metagraph visualization.

Visualization results for random metagraphs are presented in Figs. 4 and 5.

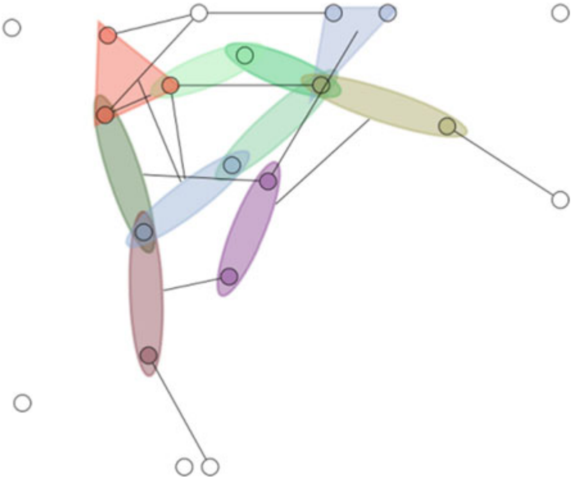


Fig. 4 Metagraph $S\,\langle |V| = 20, |M| = 10, |E| = 13\rangle$ is drawn in 1.9 s

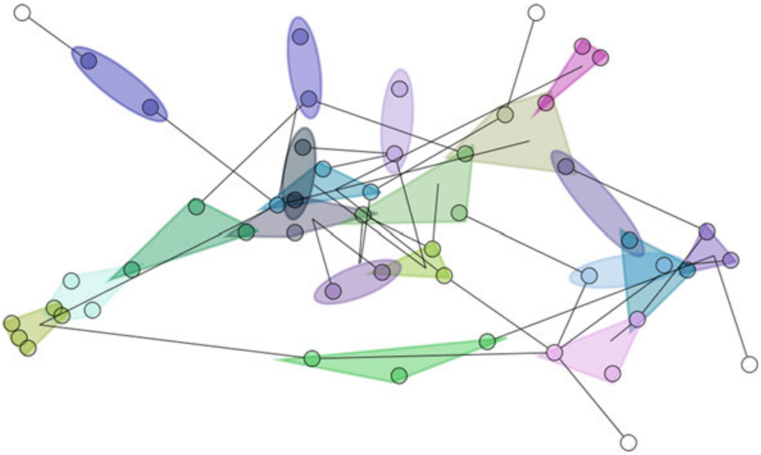


Fig. 5 Metagraph $S\ \langle |V| = 50, |M| = 20, |E| = 34\rangle$ is drawn in 10.3 s

6 Conclusion

The proposed method allows visualizing medium size metagraphs. Also there is a limit on the number of metavertices intersections. This can be explained by the fact that the Venn diagram analogue has been used. It could be difficult to discern metavertices if some of them are included in other and intersect. Several experiments were conducted that showed the usability of the proposed method.

In future revision of the method it is planned to work on the number of edges for intersection minimization, a new model of multiple metavertex intersections determination and the occupation area minimization.

References

1. Archambault, D., Munzner, T., Auber, D.: Topolayout: multilevel graph layout by topological features. IEEE Trans. Vis. Comput. Graph. **13**(2), 17–305 (2007)
2. Astanin S.V., Dragnish N.V., Zhukovsky N.K., Nested metagraphs as models of complex objects. Inž. vestn. Dona, 4 (2012), http://www.ivdon.ru/magazine/archive/n4p2y2012/1434
3. Barnes, J., Hut, P.: A hierarchical O(N log N) force-calculation algorithm. Nature **324**(4), 446–449 (1986)
4. Basu, A., Blanning, R.W.: Metagraph and Their Application. Integrated Series in Information Systems. Springer, Berlin (2007)
5. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms (2nd ed.). MIT Press and McGraw-Hill, New York (2001)
6. Eades, P.: A heuristic for graph drawing. Congr. Numerantium **42**, 149–160 (1984)
7. Frick A., Ludwig A., Mehldau H.: A fast adaptive layout algorithm for undirected graphs. In: Proceedings of the Graph Drawing 1994. LNCS, vol. 894, pp. 389–403. Springer, Berlin (1995)
8. Fruchterman, T.M.J., Reingold, E.M.: Graph drawing byforce-directed placement. Softw. - Pract. Exp. **21**(11), 1129–1164 (1991)

9. Hachul, S., Junger, M.: Drawing Large Graphs with a Potential-Field-Based Multilevel Algorithm (Extended Abstract). In: Proceedings of the Graph Drawing 2004, pp. 285–295. Springer, Berlin (2005)
10. Harel, D., Koren, Y.: LNCS. A Fast Multi-scale Method for Drawing Large Graphs. In: Proceedings of the Graph Drawing 2000. LNCS, pp. 183–196. Springer, Heidelberg (2001)
11. Holten, D.: Hierarchical edge bundles: visualization of adjacency relations in hierarchical data. IEEE Trans. Vis. Comput Graph. **12**(5), 741–748 (2006)
12. Kamada, T., Kawai, S.: An algorithm for drawing general undirected graphs. Inf. Process. Lett. **31**, 7–15 (1989)
13. Rogers, D.F.: Procedural Elements for Computer Graphics (2nd ed.). WCB/McGraw-Hill, New York (1998)
14. Sugiyama, K., Misue, K.: Graph drawing by the magnetic spring model. J. Vis. Lang. Comput. **6**(3), 217–232 (1995)
15. Walshaw, C.: A multilevel algorithm for force-directed graph drawing. J. Graph Algorithms **7**(3), 253–285 (2003)

Structural Analysis of Singularly Perturbed State Models

Walery Rogoza

Abstract The paper is devoted to the analysis of a class of mathematical models containing small parameters, namely—singularly perturbed state models. These models are of special interest for the researchers, who deal with computer simulation because the models reflect distinctive properties of the object under consideration, which should be taken into account in the practical design. Specifically, under proper conditions, small variations of parameters may cause substantial changes in physical states of the actual object. Using electronic circuits as examples, structural conditions are studied, which cause the availability of singularly perturbed state models presented in the form of the set of ordinary differential equations with small parameters on derivatives of state variables. It is suggested that the proposed approach to the structural analysis of the mentioned type of mathematical models can be extended to other classes of engineering objects by invoking the well-known principle of physical analogies.

Keywords Computer simulation · Structural analysis of state modes · Singularly perturbed models · Electronic circuits

1 Introduction

Parametric models of engineering objects are frequently dealt with in different areas of the design practice. The class of models containing parameters give a special concern among the designers, because under certain conditions small parameter variations may lead to substantial changes in the object behavior. In mathematics, one of the pioneer investigators of this problem was Hadamard [1], who introduced the mathematical term the well-posed (or well-conditioned) problem. He believed that mathematical models of physical phenomena should have the properties that:

W. Rogoza (✉)
Faculty of Computer Science and Information Technology, West Pomeranian
University of Technology, Zolnierska 49, 71-210 Szczecin, Poland
e-mail: wrogoza@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_31

(1) a solution exists, (2) the solution is unique, and (3) the solution's behavior changes continuously with small variations of the initial conditions and small variations of model parameters. Problems that are not well-posed in the sense of Hadamard are termed ill-posed. Typically, an ill-conditioned problem possesses strong sensitivity of the object behavior to small variations of some parameters.

The concept of a well-posed problem was adopted on the assumption that every mathematical problem corresponding to some physical or technological problem must be well-posed. In fact, what physical interpretation can a solution have if an arbitrary small change in the data can lead to large changes in the solution? Moreover, it would be difficult to apply numerical methods to solve such problems. However, this point of view, which is natural when applied to certain time-depended phenomena, cannot be extended to all problems.

In the paper, the class of singularly perturbed models is considered, which is close to the class of ill-conditioned problems in that such models exhibit high sensitivity of the object behavior as respect to small variations of parameters. We will concentrate our attention on the singularly perturbed state differential equations as applied to electronic circuits. Historically, mathematical properties of this class of equations have been well studied for a long time. But we would like to draw attention on the qualitative aspects of this problem, that is, what structural properties of the object analyzed predetermine the formation of the singularly perturbed state model. Attention is drawn to the fact that the mentioned state models are caused by at least two reasons: small values of some parameters presenting in the state model and specific structural and functional features of the object to be analyzed.

2 The General Structure of the Set of State Equations of Linear Objects

For the sake of definiteness let us consider the state model of a linear electronic circuit presented in the form of the set of ordinary differential equations (ODE). As is known, the state model represented in the ODE form is used for the majority of engineering objects with lumped parameters [2]. One of the advantages of the mentioned state model is that it allows us to establish direct relationships between the values of coefficients of the state model and structural properties of the actual object under consideration. Let us consider the following state model:

$$\begin{aligned} &\text{(a)} \quad M\frac{dx}{dt} = Ax + Bu + B'\frac{du}{dt}; \\ &\text{(b)} \quad y = Cx + Du, \end{aligned} \tag{1}$$

where x is the n-dimensioned vector of state variables, u is the m-dimensioned vector of input signals, y is the l-dimensional vector of output signals, and A, B, C, D, M, and B' are matrix coefficients of corresponding dimensions. Assume that voltages across capacitances and currents over inductances are chosen as state variables x.

For the sake of definiteness, we assume that the circuit may include: independent voltage sources (denote them as E-components), dependent voltage sources (V-components) controlled by voltages v or currents i of any circuit components, independent current sources (J-components), dependent current sources (I-components) controlled by voltages v or currents i, conductances (G-components), resistors (R-components), capacitances (C-components), and inductances (L-components). In addition, open-circuited components (γ), that is components, whose conductance is zero ($G = 0$), and short-circuited components (δ), that is components, whose resistance is zero ($R = 0$) are convenient to use in order to formalize definitions of some circuit functions presented below. Such a set of components is quite sufficient for the analysis of the majority of actual electronic networks, which operate in the linear mode [3].

To form state equations, we should build the circuit structural tree T with a distinct priority of edges, namely, all edges corresponding to the E- and V-components as well as the maximum possible number of edges corresponding to the C-components are assigned to the set of branches of the T tree (denote tree branches by the T subscript, for example, C_T-branches), and all the edges corresponding to the J- and I-components as well as the maximum possible number of edges corresponding to the L-components are assigned to the set of chords N (denote chords by the N subscript, for example, L_N-chords). Let us call such a tree formed with the mentioned priority of branches the normal tree of the circuit structural graph. Thus under these conditions, the state vector x in (1) will consist of voltages across the C_T-components (sub-vector v_{CT}) and currents over the L_N-components (sub-vector i_{LN}), that is, $x = (v_{CT}, i_{LN})^T$, and the input signal vector u will consist of voltages of independent voltage sources v_E and currents of independent current sources i_J, that is, $u = (v_E, i_J)^T$, where superscript T is the transposition symbol.

It is obvious that if the circuit structural graph contains n_C capacitance edges and m_l degenerate loops that consists of E-, V-, and/or C-edges, then the normal tree T will include $n_{CT} = n_C - m_l$ C-branches and m_l C-chords [4]. And in the dual case, if the graph contains $n_L$inductances and m_c degenerate cut-sets consists of J-, I-, and/or L-edges, then the set of chords will include $n_{LN} = n_L - m_c$ L-chords and m_c L-branches of the T tree [4].

It can be also shown [3, 5] that the values of C and L components as well as parameters of the controlled sources presenting in the generate loops and cut-sets are the elements of matrices M and B' in (1) [2–4]. Thus, filling with nonzero elements of the mentioned matrices depends on the way in which capacitances and inductances are connected in the circuit.

In order to establish the physical meaning of matrix coefficients A, B, C, and D in (1), it is convenient to represent the electronic circuit in the form of an N-port, in which E-, J-, C-, and L-components of the circuit are considered as the external components connected to the ports of the N-port, as is shown in Fig. 1 [2].

It can be shown [2] that elements of matrices $A, B, C,$ and D in (1) can be determined as specific input and transfer functions with respect to ports of the N-port. We call them the particular functions of the N-port, because the mentioned functions can be determined with appropriate modifications of ports of the N-port. This fact can be

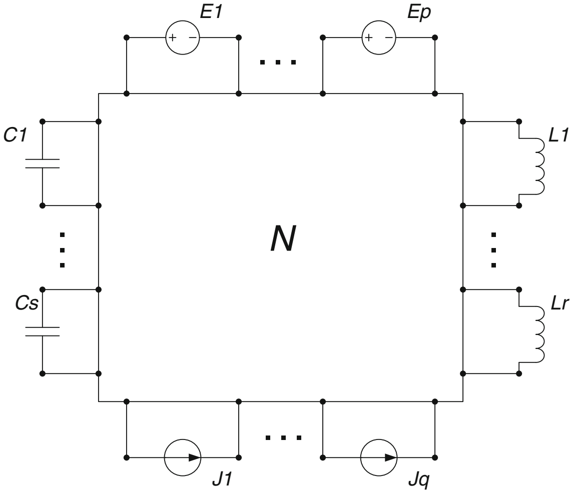


Fig. 1 The circuit as an *N*-port

reflected in the following form of state equations (1):

$$
\begin{aligned}
\text{(a)} \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \cdot \frac{d}{dt} \begin{bmatrix} v_{CT} \\ i_{LN} \end{bmatrix} &= \begin{bmatrix} Y(C,C) & K_i(L,C) \\ K_v(C,L) & Z(L,L) \end{bmatrix} \cdot \begin{bmatrix} v_{CT} \\ i_{LN} \end{bmatrix} \\
&+ \begin{bmatrix} Y(E,C) & K_i(J,C) \\ K_v(E,L) & Z(J,L) \end{bmatrix} \cdot \begin{bmatrix} v_E \\ i_J \end{bmatrix} \\
&+ \begin{bmatrix} B_1' \\ B_2' \end{bmatrix} \cdot \frac{d}{dt} \begin{bmatrix} v_E \\ i_j \end{bmatrix},
\end{aligned}
$$

$$
\text{(b)} \begin{bmatrix} v_Y \\ i_\delta \end{bmatrix} = \begin{bmatrix} K_v(C,\gamma) & Z(L,\gamma) \\ Y(C,\delta) & K_i(L,\delta) \end{bmatrix} \cdot \begin{bmatrix} v_{CT} \\ i_{LN} \end{bmatrix} + \begin{bmatrix} K_v(E,\gamma) & Z(J,\gamma) \\ Y(E,\delta) & K_i(J,\delta) \end{bmatrix} \cdot \begin{bmatrix} v_E \\ i_J \end{bmatrix}, \quad (2)
$$

where the vector of output voltages and currents of the circuit takes the form: $y = (v_\gamma, i_\delta)^T$, here, γ denotes the open-circuited output ports, and δ denotes the short-circuited output ports. As can be concluded from (2), *Y*(,), *Z*(,), K_i(,), and K_v(,) are submatrices of particular input or transfer admittance functions, particular input or transfer impedance functions, and also particular voltage transfer functions, and particular current transfer functions, respectively, determined at to the ports of the *N*-port. Any particular function is determined with respect to their own input and output ports with short-circuiting *E*-, C_T-, and L_T-branches and open-circuiting *J*-, C_N-, and L_N-chords, except of the pair of input and output ports of the given particular circuit function. The latter are denoted in parentheses: the first symbol denotes the input port, and the second symbol denotes the output port. For example, K_v (C_i, L_j) denotes the particular voltage transfer function, whose input port is the pair of nodes, to which a C_i-component is connected, and output port is the pair of nodes, to which an L_j-component is connected. In the case of a particular input function, such as $Y(\ ,\ )$, $Z(\ ,\ )$, input and out ports are the same. For example, $Y(C_i, C_i)$ denotes the particular input admittance function at the nodes, to which C_i-component is connected, and $Z(L_j, L_j)$ denotes the particular input impedance function at the nodes, to which L_j-component is connected.

The possibility of computation of any particular circuit function using techniques of the matrix algebra is of fundamental importance in our study. Let us consider this issue in detail.

3 The Algebraic Method of Computing Particular Functions

As is known [3, 5], using Laplace transformation, we can describe the behavior of a linear electronic circuit in the form of the following matrix equation:

$$Y_n v_n = J_n, \tag{3}$$

where v_n is the vector of nodal voltages of the circuit, J_n is the vector of nodal currents of independent current sources, and Y_n is the circuit nodal admittance matrix. The only restriction imposed on the model (3) is that it allows us to analyze linear circuits containing components of the admittance type, only. But this restriction is not a fundamental importance because using well-known equivalent circuit transformations [3] we can represent any circuit component as a component of the admittance type. Moreover, we can use algebraic techniques described below to compute any particular circuit function using cofactors of matrix Y_n. The following notation of the matrix Y_n cofactors will be applicable in our analysis.

$\Delta_{(i+j)(k+l)}$ is the cofactor of the Y_n matrix, in which the ith row is added to the jth row, following which the ith row is deleted, and the kth column is added to the lth column, following which the kth column is deleted.

$\Delta_{(i+j)(k+j)} \equiv \Delta_{[i+j]}$ is the cofactor of matrix Y_n, in which the indexes of summarized and deleted rows are the same as the indexes of summarized and deleted columns. It should be noted that the operations denoted in square brackets $\Delta_{[i+j]}$ are equivalent to short-circuiting nodes i and j in the actual circuit.

$\Delta_{(a+b)(c+d),[e+f],\ldots,[g+h]}$ is the summarized cofactor derived from the Y_n matrix by successive summarizing and deleting rows and columns according to the above rules.

It is a remarkable fact that any circuit function can be computed using appropriate cofactors of the circuit Y_n matrix [6]. Indeed, let α and β be some ports of the circuit. Then using cofactors of the Y_n matrix, we can write the following generalized expressions for different circuit functions.

1. The input admittance at the α port: $Y(\alpha) = -\dfrac{\Delta}{\Delta_{[\alpha]}}$.
2. The transfer admittance from the α port to the β port: $Y(\alpha, \beta) = -\dfrac{\Delta_{(\alpha)(\beta)}}{\Delta_{[\alpha],[\beta]}}$.

3. The input impedance at the α port: $Z(\alpha) = -\dfrac{\Delta_{[\alpha]}}{\Delta}$. (4)

4. The transfer impedance from the α port to the β port: $Z(\alpha, \beta) = \dfrac{\Delta_{(\alpha)(\beta)}}{\Delta^0}$.

5. The voltage transfer function from the α port to the β port: $K_v(\alpha, \beta) = \dfrac{\Delta_{(\alpha)(\beta)}}{\Delta_{[\alpha]}}$.

6. The current transfer function from the α port to the β port: $K_I(\alpha, \beta) = \dfrac{\Delta_{(\alpha)(\beta)}}{\Delta_{[\beta]}}$.

In (4), Δ denotes the matrix Y_n determinant, and the rest of notations were commented above. As was mentioned above, there are algebraic relationships, which allow us to use the Y_n matrix for the analysis of circuits with any types of components. A possible method is based on the use of expansions of the matrix Y_n cofactors in parameters of arbitrary dimensions. To illustrate this thesis, we can present expansions in parameters of four types of dependent voltage and current sources: voltage controlled current source (VCCS), voltage controlled voltage source (VCVS), current controlled current source (CCCS), and current controlled voltage source (CCVS).

With this aim, assume that the circuit includes a dependent source connected to nodes q and r in such a manner that the voltage vector is directed from the r node to the q node (it means that current flows in the opposite direction, that is, from the q node to the r node), and the controlling component is connected to nodes p and t in such a manner that the voltage vector is directed from the t node to the p node (it means that the current flows in the opposite direction—from the p node to the t node).

Assume next that we want to represent a $\Delta(\,*\,)$ cofactor as a function of a certain controlling parameter $(\,*\,)$ of the dependent source. The matrix algebra [6] offers the following expansions in parameters of any of four types of dependent sources:

$$\begin{aligned}
&\text{(a) VCCS}: \Delta(s) = \Delta && + s\Delta_{(q+r)(p+t)},\\
&\text{(b) VCVS}: \Delta(m) = \Delta_{[q+r]} && + m\Delta_{(q+r)(p+t)},\\
&\text{(c) CCCS}: \Delta(n) = \Delta_{[p+t]} && + n\Delta_{(q+r)(p+t)},\\
&\text{(d) CCVS}: \Delta(r) = \Delta_{[q+r],[p+t]} && + r\Delta_{(q+r)(p+t)},
\end{aligned} \tag{5}$$

where s, m, n, and r are the controlling parameters of the above dependent sources.

The same expansions exist for any other circuit component, and no matter what dimension it has. On closer inspection of physical conditions under which the determinants in (4) are computed to obtain any particular circuit function, we can assure ourselves that the determinants of all the mentioned functions are the same. For example, Δ^0 and $\Delta_{[\alpha]}$ are determined when all the ports, to which E-, C_T-, and L_T-branches are connected, should be short-circuited (but this rule is true for the α port, too!). We can make the same conclusion comparing any other pair of determinants in expressions (4) being applied to our particular circuit functions.

The above analytical method of computation of particular circuit functions places at our disposal a tool for the structural investigation of a class of singularly perturbed problems.

4 The Analytically Based Structural Analysis of a Singularly Perturbed State Model

A special interest to the singularly perturbed state equations is dictated by their specific features [7]. One of them is that under proper conditions small variations in the coefficients of derivatives of state variables may lead to essential variations in the behavior of the object analyzed. Does it mean that an object exhibiting such a behavior possess particular structural and functional features? To answer this question, we can use the algebraic approach described above.

Taking into account the fact that relationships (4), as applied to particular circuit functions, are different in nominators, and have the same denominators (denote the latter by Δ^0), we can rewrite matrices A, B, C, and D as follows:

$$A = \frac{1}{\Delta^0} A_1, \; B = \frac{1}{\Delta^0} B_1, \; C = \frac{1}{\Delta^0} C_1, \; D = \frac{1}{\Delta^0} D_1, \tag{6}$$

where $1/\Delta^0$ is the common scalar coefficient for the matrices give in (6).

Considering that, by definition, the inverse matrix $M^{-1} = \text{adj}M/\det M$, where adj$M$ is the adjacent matrix and det M is the determinant of matrix M, and assuming that both det M and Δ^0 are nonzero, we can rewrite (1) as follows:

$$\text{(a) } \Delta^0 \cdot \det M \cdot \frac{dx}{dt} = \text{adj}M \cdot A_1 \cdot x + \text{adj}M \cdot B_1 \cdot u + \text{adj}M \cdot \Delta^0 \cdot B' \cdot \frac{du}{dt};$$

$$\text{(b) } \Delta^0 \cdot y = C_1 \cdot x + D_1 \cdot u, \tag{7}$$

where scalar values Δ^0 and det M can be interpreted as parameters.

Considering (1) along with (7), we can conclude that there are at least two reasons why Eqs. (1) and (7) may have small parameters on state derivatives (that is, (1) or/and (7) become singularly-perturbed sets of equations): (1) det $M \to 0$ and/or (2) $\Delta^0 \to 0$. Examples discussed below illustrate physical conditions, when the mentioned relationships take place.

Example 1. There is no difficult to state conditions when det $M \to 0$, namely: small values of reactive circuit components (capacitances and/or inductances), which are present in matrix M, and/or specific relationships between the reactive components and the parameters of controlled sources. As an example, let us consider the electronic circuit (Fig. 2a) containing a negative differential resistance, which is realized using an impedance converter with current inversion (GICCI) [8].

The simplest model of the GICCI (Fig. 2b) consists of the current controlled current source (CCCS), whose current is determined by the equation $I = nI_{in}$, where

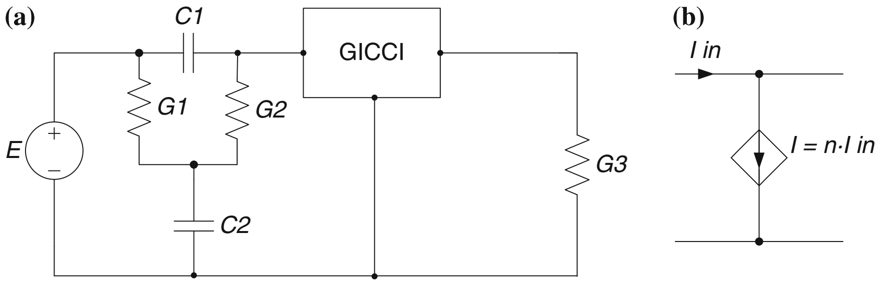


Fig. 2 **a** A circuit containing the generalized impedance converter with current inversion (GICCI). **b** The simplest model of the GICCI

I_{in} is the input current of the GICCI and n is the controlled parameter. Generally, the n parameter is determined by the two inherent parameters of the GICCI—the input z_α and the output z_β impedances (they are omitted in our illustrative example for the sake of simplicity).

The behavior of the circuit shown in Fig. 2a can be described by the following state equations:

$$\begin{bmatrix} C_1(1-n) & 0 \\ 0 & C_2 \end{bmatrix} \frac{d}{dt} \begin{bmatrix} v_{C1} \\ v_{C2} \end{bmatrix} = \begin{bmatrix} (n-1)G_2 - G_3 & (n-1)G_2 \\ -G_2 & -(G_1+G_2) \end{bmatrix} \begin{bmatrix} v_{C1} \\ v_{C2} \end{bmatrix} + \begin{bmatrix} (1-n)G_2 + G_3 \\ G_1 + G_2 \end{bmatrix} v_E, \qquad (8)$$

where v_{c1}, v_{c2}, and v_E are voltages of components C_1, C_2, and E, respectively, and $n = 1 - z_\beta / z_\alpha$.

The relationship between z_α and z_β governs the result of conversion, namely: if $z_\beta > z_\alpha$, then n is negative and the GICCI operates as an impedance converter, but if $z_\beta \ll z_\alpha$, then $n \to 1$ and the coefficient of dv_{c1}/dt on the left side of (8) approaches zero. This situation corresponds to the above case when the determinant at the left side of state equations approaches zero (det $M \to 0$).

As can be seen, if we set $n = 1$, then the first equation in (8) is transformed to the algebraic equation:

$$av_{c1} + bv_{c2} + cv_E = 0, \qquad (9)$$

where $-a = c = G_3$ and $b = 0$. In other words, voltages are related by the linear relationship of much as if it would be in the case of a degenerate loop comprised of C_1, C_2, and E components. It should be noted that in this situation, the reduced set of differential equations occurs not as a consequence of small magnitudes of circuit parameters, but because of a certain relationship between the values of the matrix M elements.

Note, please, that in the simplest case, det $M \to 0$ if $C_1 \to 0$ and/or $C_2 \to 0$. In any case, zeroing small parameters, we can obtain the reduced form of state equations, whose solution $\bar{x} = (\overline{v_{c1}}, \overline{v_{c2}})^T$ (here, T is the transposition symbol) approaches the solution of the perturbed equations (1) $x = (v_{c1}, v_{c2})^T$ only if certain conditions are met. These conditions follow directly from the general mathematical representation of the solution of the perturbed equations (8):

$$\begin{aligned} &\text{(a) } v_{c1}(t) = \overline{v_{c1}} + k_1 \exp\left(\frac{\lambda_1 t}{\mu}\right) + k_2 \exp\left(\frac{\lambda_2 t}{\mu}\right); \\ &\text{(b) } v_{c2}(t) = \overline{v_{c2}} + k_1 \exp\left(\frac{\lambda_1 t}{\mu}\right) + k_2 \exp\left(\frac{\lambda_2 t}{\mu}\right), \end{aligned} \tag{10}$$

where λ_1 and λ_2 are the roots of the characteristic equation derived from (8), and k_1, k_2 are some constants.

As can be seen, solutions $\overline{v_{c1}}$ and $\overline{v_{c2}}$ of the reduced equations approach solutions $v_{c1}(t)$ and $v_{c2}(t)$ of the perturbed equations if λ_1 and λ_2 are negative and are great enough. This condition imposes certain limitations on values of circuit components. For example, $n \le 1$ is one of them.

If limitations are met, then with a reasonably small μ we can observe two time regions, within which the processes described by (10) have substantially different rates—the boundary layer, as it is called, with relatively fast rates of variable variations and the region beyond the boundary layer with relatively slow rates of variable variations. The availability of one or more boundary layers is typical for the solution of any singularly perturbed set of differential equations [9].

Example 2. This example shows the second case of the singularly perturbed state model mentioned above. Let us consider the circuit shown in Fig. 3. The circuit consists of the independent voltage source *E,* voltage controlled voltage source *V*, whose voltage is controlled by voltage of *E* and is determined by the equation $v_V = mv_E$ (where m is the controlling parameter), inductances L_1 and L_2, capacitance C_1, and conductances $G_1 – G_5$.

In order to compute the common denominator Δ^0 of particular circuit functions, it is required to form the normal tree of the circuit structural graph, next C_T and L_T tree branches should be short-circuited and C_N and L_N chords should be removed from the graph. Following these procedures, namely, on short-circuiting C_1 and removing L_1 and L_2, we can use Eq. (5b) to compute the Δ^0 value as a function of the m parameter of the VCVS. In our case, the numbers of nodes are as follows:

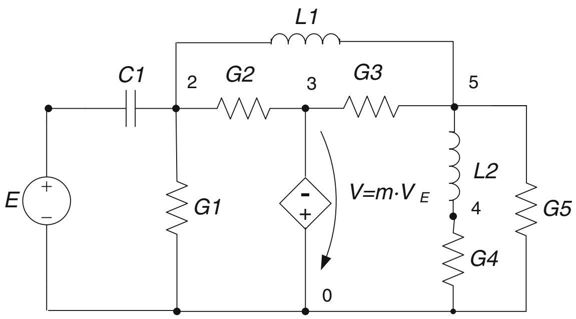


Fig. 3 A circuit containing a dependent source

$q = t = 0$, $r = 3$, and $p = 1$. Consequently, the required Δ^0 value can be computed using the equation:

$$\Delta^0(m) = \Delta^0_{[1+2],[0+3]} + m\Delta^0_{[1+2],(0+3)(1+0)} = \Delta^0_{[1+2],[3+0]} - m\Delta^0_{[1+2],(3+0)(2+0)}$$
$$= (G_1 + G_2)(G_3 + G_5)G_4 - mG_2(G_3 + G_5)G_4.$$

Thus $\Delta^0(m) \to 0$ if $m \to (G_1 + G_2)/G_2$. It should be noted once more that this relationship (as well as in our previous example) is met for nonzero circuit components and does not suggest their small values.

5 How Are Ill-Posed Problems and Singularly Perturbed Problems Alike and How Do They Differ?

A strong sensitivity of some output variables of the analyzed object to small variations of parameters is the common feature of the ill-posed problems and singularly perturbed problems. However ill-posed problems hold a continuous dependence of output variables within the entire interval of parameter variations, whereas in the case of singularly perturbed problems the mentioned continuous dependence may be broken down when the values of some parameters reach zero—the factor that is sometimes ignored by the designers. We would like to mention briefly this issue because it is closely allied to the subject matter of our discussion.

The mathematical properties of singularly perturbed sets of ODE have been studied for a long time by a number of researchers. A. Tikhonov was one of them (for example, see [9]). He studied the Cauchy problem of the form:

$$\begin{cases} \text{(a) } \mu\dot{x} = f(x, y), x(0) = x^0, x \in \mathbf{R}^n, \\ \text{(b) } \dot{y} = g(x, y), y(0) = y^0, y \in \mathbf{R}^m, \end{cases} \tag{11}$$

where x and y are the n-dimensional and m-dimensional sub-vectors of state variables, respectively, determined in real spaces, and μ is the matrix of parameters, whose magnitudes are small enough (say, with respect to unity). The initial conditions for state variables x and y are given by the x^0 and y^0 vectors. Following Tikhonov's theory, Eq. (11a) exhibits processes with relatively fast rates and Eq. (11b) processes with relatively slow rates.

For reasonably small values of parameters μ the set of equations (11) possesses the stiffness property, what complicates their numerical solution [10]. A way to simplify the problem is to set small parameters μ to zero. Following such a simplification, model (11) can be reduced to the form of unstiff equations:

$$\begin{cases} \text{(a) } 0 = f(x, y), x \in \mathbf{R}^n, \\ \text{(b) } \dot{y} = g(x, y), y(0) = y^0, y \in \mathbf{R}^m, \end{cases} \tag{12}$$

which can be solved using simpler in a certain sense numerical methods. In this connection, three remarks are to be made.

First, a great number of examples can be mentioned, which show that model (11) exhibits some specific features of the object analyzed, whose ignoring may be intolerable in the design. For example, very-large-scale integrated circuits (VLSI) provide a good example of those complex engineering objects, whose small model parameters exhibit various second-order effects as they are called [11]. The analysis of these effects may be one of the goals of computer simulation, and therefore such parameters should not be ignored.

Second, as was shown in the above discussion, small coefficients on some derivatives of state variables may be caused not only by small values of some parameters, but also because of the particular relationships between component values of the object analyzed. Typically, the mentioned relationships point to some kind of functional dependences between the parts of a complex object, which require closer examination in each case. Say, lateral transistor effects between the components which are mounted close together on the substrate of a VLSI offer examples of this kind relationships.

And third, because of the fact that in the reduced set of equations (12) we ignore initial values for variables x, the solution of (12) need not be approached to the solution of the perturbed set of equations (11), and sometimes these solutions may be as far as is wished. According to Tikhonov's theory [9], if the solution $\bar{x} = \varphi(y)$ of (12a) exists and is continuous with variations of variables y, and on substitution this solution into the second matrix equation (12b) we obtain the solution of (12) that approaches the solution (x, y) of the perturbed equations (11), then $\bar{x} = \varphi(y)$ is called the stable root of the reduced equation (12a). Hence, if we use a stable root of (12a) and the vector (x^0, y^0) of initial conditions belongs to the attraction domain D of the stable root within the observation time interval $t \in [0, T]$, then the solution of the perturbed equations (11) approaches asymptotically the solution of the reduced equations (12), what can be written as follows:

$$\begin{aligned} x(t,\mu) &\to \bar{x}(t), \tau \le t \le T, \\ y(t,\mu) &\to \bar{y}(t), 0 \le t \le T, \end{aligned} \tag{13}$$

where it is assumed that $\mu \to +0$ and $(\bar{x}, \bar{y})$ is the solution of the reduced problem (12), and $[0, \tau]$ is a relatively narrow boundary layer within which "fast" processes largely take place.

But if the above conditions are not met, the solution of the reduced set of equations (12) and the perturbed set of equations (11) may be far from each other as much as you like. The following simple example illustrates this fact.
Example 3. Let us consider the following singularly perturbed equation:

$$\mu \frac{dx}{dt} = x(t^2 - x + 1), x(t_0) = x_0, \tag{14}$$

where $\mu > 0$ is a small parameter.

The reduced equation obtained by setting $\mu = 0$ is as follows: $0 = x(t^2 - x + 1)$. This equation possesses two roots: (1) $x = 0$ and (2) $x = t^2 + 1$. Intuition suggests that the two roots provide different results of approximation. But which root should

Table 1 Analogies between physical variables

Electrical variable	Mechanical variable	Acoustic variable
Inductance, L	Mass, m	Acoustic inertness, M
Electrical charge, q	Linear displacement, x	Volume displacement, X
Time, t	Time, t	Time, t
Current, i	Linear velocity, v	Solid current, U
Electromotive force, e	Force, f_M	Sound pressure, p
Electrical resistance, r	Mechanical resistance, r_M	Acoustic resistance, r_A
Electrical capacitance, C	Pliability, C_M	Acoustic capacitance, C_A

be selected? According to Tikhonov's theory [9], the first root is unstable because $\frac{\partial[x(t^2-x+1)]}{\partial x} = t^2 + 1 > 0$ for $x = 0$, and the second root is stable because $\frac{\partial[x(t^2-x+1)]}{\partial x} = -t^2 - 1 < 0$ for $x = t^2 + 1$.

Hence, if the initial point (t_0, x_0) lies in the upper half-plane (that is, $x > 0$), then the curve of the solution of (14) approaches smoothly the $x = t^2 + 1$ curve. But if the initial point (t_0, x_0) lies in the lower half-plane, then the curve of the solution of the perturbed equation (14) and the curve of the solution of the reduced equation are moving farther apart.

It is evident that the mentioned problem may be the subject of special studies, whose aim is the development of special-purpose numerical methods of solution of singularly perturbed sets of ODE. One of such methods has been proposed in [12].

And another question is as follows: is it possible to expand the ideas of structural analysis, discussed in the paper on other classes of objects different from the class of electronic circuits studied above? To confirm this suggestion, we would like to mention a well-known principle of physical analogies. As is known, there are certain correspondences between variables in different physical areas. By way of illustration, Table 1 represents analogies between electrical, linear mechanical, and acoustic variables.

Hence, taking into account formal analogies between physical variables, it is safe to assume that the development of the above approach in other engineering areas is valid.

6 Conclusion

Analyzing properties of mathematical models, designers make decisions allowing them to enhance characteristics of objects and to avoid crucial physical states, which can entail unforeseen changes in the object behavior. If the designer deals with the singularly perturbed state equations, then the mentioned changes can be caused by specific structural and physical features of the analyzed object. Therefore it seems reasonable to say that investigations of problems related to the analysis of singularly perturbed state models as well as traditional ill-conditioned problems should occupy much attention in the computer simulation especially if we deal with complex objects.

It is anticipated that using the principle of physical analogies, the structural analysis of the mentioned models can be carried out to other classes of engineering objects, say, mechanical and acoustic objects.

Consequently, the study of physical conditions, which are responsible for the existence of the singularly perturbed models, may be of specific interest in the investigation of critical states of systems of different kinds as well as in the design of robust engineering objects.

References

1. Hadamard, J.: Sur les problèmes aux dérivées partielles et leur signification physique. Princeton Univ. Bull. **13**, 49–52 (1902)
2. DeRusso, P.M., Roy, R.J., Close, C.M., Desrochers, A.D.: State Variables for Engineers. Wiley, New York (1997)
3. Desoer, C., Kuh, E.: Basic Circuit Theory. McGraw-Hill, New York (1969)
4. Sommariva, A.M.: State-space equations of regular and strictly topologically degenerate linear lumped time-invariant networks: the multiport method. Int. J. Circuit Theory Appl. **29**, 435–453 (2001)
5. Riaza, R.: Differential-Algebraic Systems: Analytical Aspects and Circuit Applications. World Scientific Publishing, Singapore (2008)
6. Vlach, J.: Linear Circuit Theory: Matrices in Computer Applications. Apple Academic Press, Toronto (2014)
7. Kadalbajo, M.K., Patiodar, K.C.: A survey of numerical techniques for solving singularly-perturbed ordinary differential equations. Appl. Math. Comput. **130**, 457–510 (2002)
8. Huelsman, L.P.: Active and Passive Analog Filter Design. An Introduction, p. 480. McGraw-Hill Co., New York (1993)
9. Tikhonov, A.N.: Systems of differential equations containing a small parameter multiplying the derivative. (in Russian). Mat. Sb. **31**(73), 575–586 (1952)
10. Brugnano, L., Trigiante, D.: On the characterization of stiffness for ODEs. Dynam. Contin. Discrete Impulse Syst. **2**(3), 327–335 (1996)
11. Ferry, D.K., Akers, L.A., Greeneich, E.W.: Ultra Large Scale Integrated Microelectronics, p. 316. Prentice Hall, Englewood Cliffs (1988)
12. Rogoza, W.: Adaptive simulation of separable dynamical systems in the neural network basis. In: Pejas, J., Piegat, A. (eds.) Enhanced Methods in Computer Security, Biometric and Artificial Intelligence Systems, pp. 371–386. Springer, New York (2005)

The Formal Model of Data Mining Algorithms for Parallelize Algorithms

Ivan Kholod, Zaynidin Karshiyev and Andrey Shorov

Abstract The present paper describes the formal model of data mining algorithms. These models consider each data mining algorithm as a sequence of operations. This allows us to determine ways for parallel execution of data mining algorithms. The software implementation of the formal model is executed on the Java language. A few data mining algorithms were developed on the basis of the suggested formal modal. The algorithm k-means is described in the paper as the example.

Keywords Data mining · Parallel data mining · Data mining algorithms · Parallel algorithms

1 Introduction

At present multicore processors are widely used all over the world. They can be for execution of parallel programs. Unfortunately, the algorithms are modified for parallel execution very slowly, because it is quite a difficult task. Data mining algorithms are no exception. The task of parallelizing data mining algorithm is very important, because these algorithms are often used only for analyzing large data. Therefore, it is very important to increase their performance by means of algorithm parallelization. We suggest a formal model, which makes it possible to consider a data mining algorithm as a sequence of operations and structural elements. This sequence can be transformed for parallel execution by means of adding a special structural element.

I. Kholod (✉) · Z. Karshiyev · A. Shorov
Saint Petersburg Electrotechnical University "LETI", Ul. Prof. Popova 5,
Saint Petersburg, Russia
e-mail: iiholod@mail.ru

Z. Karshiyev
e-mail: zaynidin85@gmail.com

A. Shorov
e-mail: ashxz@mail.ru

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_32

The implementation of the formal model has been performed as a set of classes on Java language. The classes were used for implementation of a few data mining algorithms. We will describe the algorithm k-means as an example.

Related works are doing other researchers [1, 2]. However, these works explore implementation parallel data mining algorithms and they do not use any formal theory for it. Furthermore, they use only MapReduce platform for implementing distributed execution of data mining algorithms.

2 Formal Model of Data Mining Algorithm

In order to formally describe a data mining algorithm, first we are going to consider the general concept of their performance (Fig. 1) [3, 4].

A data mining algorithm takes an input dataset and creates an output mining model. Thus, a data mining algorithm can represent as a function with the dataset as function argument and constructed mining model as returned value:

$$F : D \rightarrow M$$

Input dataset can present as triple [5]:

$$D = \langle A, T, W \rangle$$

where A—attributes of the dataset:

$$A = \{a_1, \ldots, a_i, \ldots, a_m\}$$

where each attribute a_i has self the range of values $D(a_i)$.

T—target attributes of the dataset:

$$T = \{a_{m+1}, \ldots, a_k, \ldots, a_t\}$$

W—set of data vectors:

$$W = \{w_1, \ldots, w_r, \ldots, w_z\}$$

where each vector w_r is set of values of the attributes:

$$w_r = \{v_{1.r}, \ldots, v_{i.r}, \ldots, v_{m.r}, v_{m+1.r}, \ldots, v_{k.r}, \ldots, v_{m+t.r}\}$$

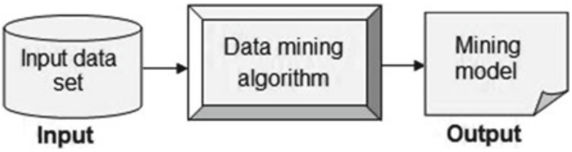


Fig. 1 General concept of data mining algorithm performance

where $v_{i.r} \in D(a_i)$—value of attribute a_i of the vector of w_r.

Thus, pair *(A, T)* is a metadata of input dataset.

Different data mining algorithms build different mining models. The paper [5] describes that basic mining models can be presented as unified mining model. This model is represented as pair:

(1) $$M = \langle R, K \rangle$$

where *R*—set of classification rules:

$$R = \{r_1, \ldots, r_i, \ldots, r_n\}$$

each rule has the conditional part and the concluding part. The rule is valid if it has both parts and each part is valid. The mining model is valid if all of its rules are valid.

K—the set of the numbers of vectors belonging to classes and clusters of the input dataset:

$$K = \{k_1, \ldots, k_l, \ldots, k_g\}$$

where k_l—number of vectors belonging to *l*-st class or cluster of input dataset.

Data mining algorithm discretely changes a mining model, i.e., the mining model can have one valid state at one moment:

- M_0—initial state of the mining model;
- M_1—mining model after the first change;
- ………
- M—final mining model;

In this case, the algorithm can represent as a sequence of executed operations:

$$F = \{O_1, \ldots, O_p, \ldots, O_q\}$$

where O_p—operation (block) of the algorithm, which changes the mining model based on the input dataset. Changed mining model must be passed to the next operation.

Thus, operation can represent as function with two arguments: input dataset and mining model before change. Function's result is mining model after change:

$$O_p : (D, M_{p-1}) \rightarrow M_p$$

Since mining model must be valid, operation must execute whole action and return valid mining model.

Thus, a data mining algorithm can be represented as an ordered sequence of operations; the result (mining model) of each operation is transferred to the following operation in sequence as an argument:

$$F = \{O_1(D, M_0), O_2(D, M_1), \ldots, O_p(D, M_{p-1}), \ldots, O_q(D, M_{q-1})\}$$

Each operation $O_p(D, M_{p-1})$ can also be decomposed into other operations which are executed sequentially:

$$O_p(D, M_{p-1}) = \{O_{p.1}(D, M_{p-1}), O_{p.2}(D, M_{p-1.2}), \ldots, O_{p.y}(D, M_{p-1.y-1})\}$$

Since the mining model can represent as a set of classification rules (1) then all operations can split to three categories:

- add operations O^{add}—operations of this category add new rules into a mining model;
- delete operations O^{del}—operations of this category delete rules from a mining model;
- change operations O^{ch}—operations of this category change a mining model's rules.

In additional to operation changing mining model, the data mining algorithms can contain structural elements: conditional statements, cycles (including cycles by vectors and by data [6]), and parallel branches. We present these structural elements as complex operations.

The conditional operation can represent as element by the following view:

$$O_{cond}(D, M_{p-1}) = \{d(D, M_{p-1}), O_t(D, M_{p-1}), O_f(D, M_{p-1})\}$$

where $d(D, M_{p-1})$—the conditional function, which returns true or false value, based on the analysis of the input dataset D and the current mining model M_{p-1}, but it does not change them;
$O_t(D, M_{p-1})$—the operation, which will execute if conditional function $d(D, M_{p-1})$ will return true;
$O_f(D, M_{p-1})$—the operation, which will execute if conditional function $d(D, M_{p-1})$ will return false.

Cycle can represent as element by the following view:

$$O_{cycle}(D, M_{p-1}) = \{d(D, M_{p-1}), O_c(D, M^*_{p-1})\}$$

where $O_c(D, M^*_{p-1})$—operation, which will execute while conditional function $d(D, M_{p-1})$ will return true;
M^*_{p-1}—mining model is changed by operation O_c on eache iteration of cycle.

Most data mining algorithms have cycle where each data vector w_r is analyzed or each attribute a_i is analyzed. For them, we add special structural elements: cycle for vectors and cycle for attributes. We present them by the following view:

$$O_{vect}(D, M_{p-1}) = \{d(W), O_c(D, M^*_{p-1})\}$$

$$O_{attr}(D, M_{p-1}) = \{d(A), O_c(D, M^*_{p-1})\}$$

where *d(W)*—the conditional function, which checks whether all the vectors have been considered or not;
d(A)—the conditional function, which checks whether all the attributes have been considered or not;

Parallel branches can represent by the following view:

$$O_{parall}(D, M_{p-1}) = \{[O_d(D, M_{p-1})], O_1(D, M_{p-1}), \ldots, O_h(D, M_{p-1})\}$$

where

$O_d(D, M_{p-1})$—dispatcher, it manages other operations $O_1 \ldots O_{h.}$ If $O_d(D, M_{p-1}) = \emptyset$ means parallel branches will be executed without a dispatcher.
$O_1 \ldots O_h$—operations are executed in parallel branches. If $O_1 = \cdots = O_h$ means it is data parallelism else task parallelism.

Thus, the data mining algorithm can contain both elements: operations changing mining model and structural elements. We generalized them in term—algorithm's block $b_p(D, M_{p-1})$. Thus, a data mining algorithm can present as a sequence of algorithm's block:

$$F = \{b_1(D, M_0), \ldots, b_p(D, M_{p-1}), \ldots, b_q(D, M_{q-1})\}$$

where $b_p \in \{O^{add}, O^{del}, O^{ch}, O_{cond}, O_{cycle}, O_{vect}, O_{attr}, O_{parall}\}$.

3 The Program Implementation of Formal Model

We extended Xelopes library [7] (data mining algorithms library) and implemented all described types of algorithm's block as classes of an object-oriented language Java. Figure 2 shows the class diagram of these blocks. Here the operation changing mining model $O_p(D, M_{p-1})$ is described by the class Step.

The implementation actions of the algorithm are contained in the method *execute()*. Calling of these methods leads to execution of the step. This method returns mining model and has followed input arguments:

- constructing mining model—*model*;
- dataset—*inputData*.

The method *execute()* is the basic step. Therefore, it must be implemented as the pure function. Since, it works only with input arguments (*model* and *inputData*) and has not any references to other variables.

To determine the sequences of blocks should be used objects of the class *SequenceOfSteps*. The sequence of blocks in itself is an algorithm's block, therefore the class *SequenceOfSteps* is inherited from the class *Step*. The object of this class contains a set of objects of class *Step* corresponding to the algorithm's block and being executed sequentially one after another.

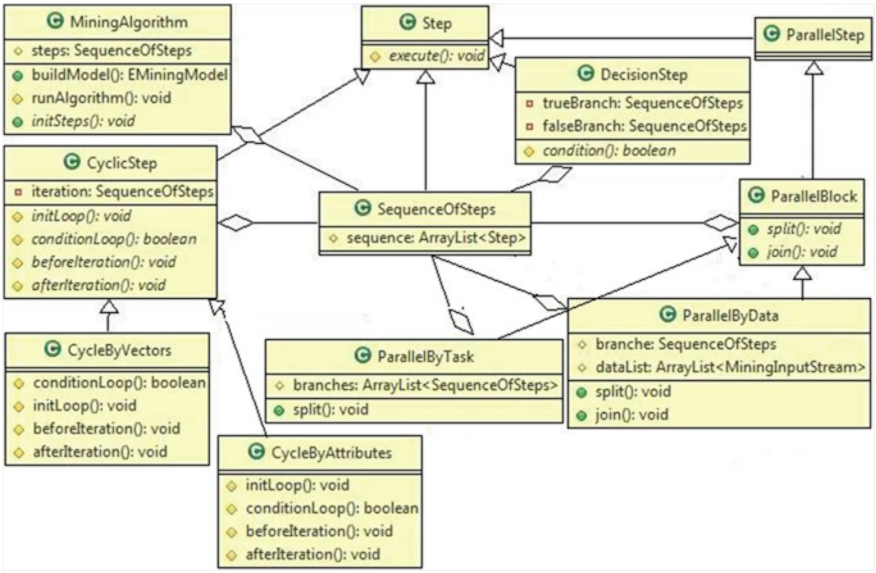


Fig. 2 Block structure of a data mining algorithm

Since the conditional operation and the cycle are steps of the algorithm, classes corresponding to them are inherited from the class *Step*. The class *DecisionStep* corresponds to the conditional operation and the class *CyclicStep* to the cycle.

Since conditional branch and the cycle are steps of the algorithm, classes corresponding to them are inherited from the class *Step*. Class *DecisionStep* corresponds to the conditional branch and *CyclicStep* to the cycle. The class *DecisionStep* in addition to the methods and attributes of the class Step contains:

- method *condition()* which implemented operation $d(D, M_{p-1})$;
- sequence of steps which executed when the condition is true—*trueBranch*, which matches to operation $O_t(D, M_{p-1})$;
- sequence of steps which executed when the condition is false—*falseBranch* which matches to operation $O_f(D, M_{p-1})$.

The class *CyclicStep* in addition to the methods and attributes of the class *Step* defines sequence of the steps, which make up one iteration of a cycle—*iteration* which matches to operation $O_c(D, M^*_{p-1})$. The condition of cycle termination $d(D, M_{p-1})$ is implemented in the method *conditionLoop()*, in addition to the methods defined in the class *CyclicStep*.

For the implementation of operation cycle for vectors $O_{vect}(D, M_{p-1})$ define the class *CycleByVectors*. It implements necessary methods of the class *CyclicStep*, providing selection of vectors of a dataset. Processing of each vector is determined by the sequence of steps added to the iteration–iteration.

Similarly, for processing of values of attributes the class *CycleByAttributes* is implemented (for operation cycle for attributes $O_{attr}(D, M_{p-1})$).

To form the target algorithm defined a class *MiningAlgorithm*. It contains a sequence of all algorithms' blocks—steps, and also methods:

- *initSteps()*—initializing steps of the algorithm;
- *runAlgorithm()*—launching the algorithm;
- *buildModel()*—building the model.

In essence in the method *initSteps()* occurs formation of algorithm structure by creating of the steps which determining of sequence and nesting of their execution.

For possibility of parallel execution of parts of algorithm the class *SequenceOfSteps* implements the interface *java.lang.Runnable*, determining thereby that the sequence of steps can be launched in a separate thread. What parts of the algorithm must be executed in separate sequences (and as a consequence in the subsequent will be executed in separate threads) is defined in the method *initSteps()* in the process of formation of the algorithm and corresponding objects of the class *SequenceOfSteps*.

For implementation parallel branches $O_{parall}(D, M_{p-1})$ implemented class *ParallelStep*. This class is the marker of parallel step. As an algorithm's block, it also inherits from the class *Step*. Main class for parallel execution is class *ParallelBlock*. It implements two main methods:

- *split()*—in this method created necessary number of branches of the parallel algorithm (by cloning of sequence of steps), and if necessary data and models are distributed among the branches;
- *join()*—in this method occurs processing of results of each branch.

Most typical type of parallelization for data mining algorithms is parallelization by data. In this case only the part of data will be sent to each parallel branch. For parallelization of algorithms by data class *ParallelByData* is added. It is inherited from the class *ParallelBlock*. In this case all parallel branches execute the same sequence of blocks, but process different data. Therefore, the new members of the class are a sequence of blocks—*branche* and a list of input data for each branch of the parallel algorithm—*dataList*. For parallelization of algorithms by task class *ParallelByData* is added. It is also inherited from the class *ParallelBlock*, but contains a list of difference sequences of blocks–branches.

4 The Example and Experiments

As an example, we will consider in more detail k-means algorithm [7] (Fig. 3)

Thus, the k-means algorithm can be described formally: as follows:

$$
\begin{aligned}
&(2)\ F=\{(d(C),O^{add}{}_{1})),\\
&\qquad ((d(CentroidsChanged),\\
&\qquad\qquad (d(W),(d(C),O^{ch}{}_{2})),\\
&\qquad\qquad (d(C),(O^{ch}{}_{3},\ O^{ch}{}_{4}))))\}
\end{aligned}
$$

```
1. for j=1 to |C |  //for each cluster from set of clusters C
2.     M←InitClusterByRandom($c_j$) // add j-st cluster with random center
3. end for j;
4. while CentroidsChanged==true
5.     for i=1 to |W | //for each vector
6.         for j=1 to |C | //for each cluster
7.             M←FindNearestClusterForVector($w_i$,$c_j$) //add  vector to nearest cluster $c_j$
8.         end for j;
9.     end for i;
10.    for j=1 to |C | //for each cluster
11.            M←ResetCenterOfClusters($c_j$) //reset center of j-st cluster
12.            M←SetCentroidOfCluster($c_j$) //recalculate center of j-st cluster
13.    end for j;
14. end while;
```

Fig. 3 The k-means algorithm

where *d(CentroidsChanged)*—the conditional function, which checks whether have been changed any the cluster's center or not;

d(C)—the conditional function, that checks whether all the clusters have been considered or not (i.e., cycle for clusters).

So need to implement follow operations and classes are implemented them:

- *InitClusterByRandom* (O_1^{add})—initialize of clusters by random way;
- *FindNearestClusterForVector* (O_2^{ch})—find nearest cluster for vector;
- *ResetCenterOfClusters* (O_3^{ch})—reset centers of clusters;
- *SetCentroidOfCluster* (O_4^{ch})—update centers of clusters;

and structures elements:

- *CycleByVector* (O_{vect})—cycle by vectors (at line 5);
- *CycleByCluster* ($O_{cluster}$)—cycle by clusters (at lines 1, 6 and 10);
- *FindClusters* ($O_{CentroidsChanged}$)—cycle while clusters are changed (at line 4).

Classes for these blocks are viewed on the Fig. 4.

The view of the k-means algorithm as a formal model (2) helps to select several approaches for parallel execution. As an example, the algorithm can execute two parallel approaches:

- run the whole algorithm on multiple processors as streams;
- parallel execute a separate block of the algorithm, which changes the state of the model and requires large computation.

For implementation of second approach need to choose the block, which can be parallel executed. The k-means algorithm contains four blocks changing model. It is necessary to choose among them the most complex block. Such block is cycle for vectors O_{vect} with operation O_2^{ch} (finding of nearest cluster for a vector).

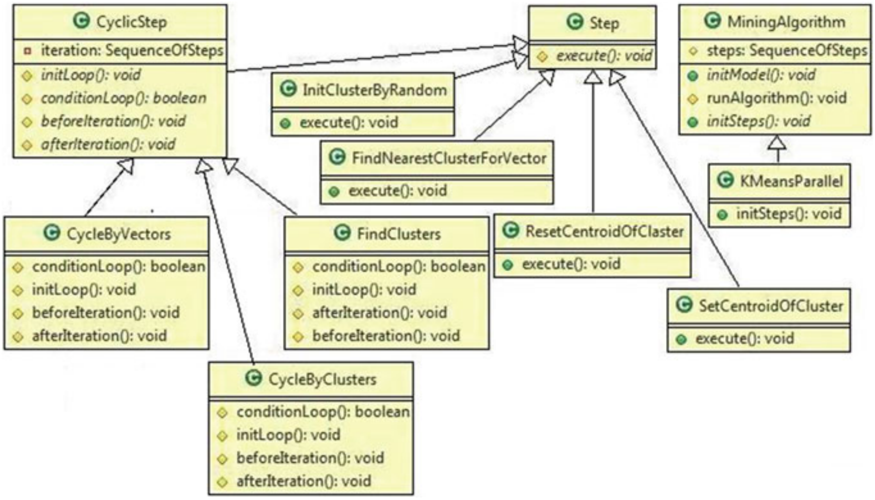


Fig. 4 The class diagram of classes for implementation of the k-means algorithm

Now consider the formal descriptions of both the approaches of parallel execution of the k-means algorithm:

- for the first approach:

$$\begin{aligned} (3)\qquad F=\{&\boldsymbol{O_d},\\ &(d(C),O^{add}{}_{1})),\\ &((d(CentroidsChanged),\\ &(d(W),(d(C),O^{ch}{}_{2})),\\ &\quad(d(C),(O^{ch}{}_{3},\ O^{ch}{}_{4}))))\} \end{aligned}$$

- for the second approach:

$$\begin{aligned} (4)\qquad F=\{&(d(C),O^{add}{}_{1})),\\ &((d(CentroidsChanged),\ (\boldsymbol{O_d},\\ &\qquad(d(W),(d(C),O^{ch}{}_{2})),\\ &\qquad(d(C),(O^{ch}{}_{3},\ O^{ch}{}_{4})))))\} \end{aligned}$$

For implementation of the sequential form (2) and both parallel forms (3) and (4) of the k-means algorithm was developed classes are viewed on Fig. 4.

We did several experiments for these implementations of the k-means algorithm. Implementation of both was easy. We added only one element of class ParallelByData (dispatcher for parallel branches $\boldsymbol{O_d}$) to method initSteps() of class KMeansParallel and inserted this element into two different places of the k-means algorithm. The experimental results are shown in Table 1. Only the second method, which employs parallel execution of single blocks allows us to reduce the execution time.

Thus, the presentation of a data mining algorithm on bases of the suggested formal model makes it possible to divide the algorithm into unified blocks. Such a

Table 1 Experimental results

Number of vectors	150	1,500	3,000	6,000
Sequential form (2) (ms)	27.834	36.918	553.27	1035.382
First approach (whole algorithm (3)) (ms)	40.893	118.984	1219.7	2760.848
Second approach (single block (4)) (ms)	19.445	27.182	341.9	676.175

splitting of data mining algorithms into blocks helps to create with less effort new algorithms from existing blocks or modify of the existing algorithms by replacing separate blocks. Additionally, this approach also allows us to easily create parallel algorithms from the sequenced algorithm by adding special structural elements for parallel execution.

Acknowledgments The work has been performed in Saint Petersburg Electrotechnical University "LETI" within the scope of the contract Board of Education of Russia and science of the Russian Federation under the contract No̲ 02.G25.31.0058 from 12.02.2013. This paper is also supported by the federal project "Organization of scientific research" of the main part of the state plan of the Board of Education of Russia and project part of the state plan of the Board of Education of Russia (task # 2.136.2014/K).

References

1. Amol, G., Prabhanjan, K., Edwin, P., Ramakrishnan, K.: NIMBLE: a toolkit for the implementation of parallel data mining and machine learning algorithms on mapreduce. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'11), pp. 334–342. California (2011)
2. Andrew, Y.Ng., Bradski, G., Chu, C-T., Olukotun, K., Kim, Sang K., Lin, Y-A., Yu, Y.: Map-reduce for machine learning on multicore. In: Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, pp. 281–288. Vancouver, Canada (2006)
3. Common Warehouse Metamodel (CWM) Specification. http://www.omg.org/spec/CWM/1.1/
4. Java Specification Request 73: Java Data Mining (JDM) - JDM Public review Draft 2003/11/25 : JSR-73 Expert Group
5. Kholod, I.I.: Unified data mining model. In: XV International Conference on Soft Computing and Measurements SCM'2012, vol. 1, pp. 237–240. Saint-Petersburg (2012)
6. Kholod, I.I., Karshiyev, Z.A.: Parallelization of the algorithm Naïve Bayes on the basis of block structure. In: XV International Conference on Soft Computing and Measurements SCM'2012, vol. 1, pp. 182–185. Saint-Petersburg (2012)
7. Barsegian, A., Kupriyanov, M., Kholod, I., Thess, M.: Analysis of Data and Processes: From Standard to Realtime Data Mining, p. 300. Re Di Roma-Verlag (2014)

Toward Generalization of Mutant Clustering Results in Mutation Testing

Anna Derezińska

Abstract Mutation testing is effectively used for evaluation of test case quality but suffers from high cost required for its realization. Mutated programs are injected with program changes specified by various mutation operators. One of the methods applied to the reduction of mutant number is mutant clustering. Instead of using all generated mutants, special mutant groups are distinguished and group representatives are used in further evaluation of tests. Mutant clustering gave some promising results for C programs. In case of object-oriented programs with standard and object-oriented operators the results were positive but not superior to other cost reduction techniques. An open issue is interpretation of mutant clustering results and their generalization to other projects in terms of used mutation operators. In this paper, three metrics are proposed to comprehend mutation clustering. Experimental results are analyzed toward usefulness of mutants created by various operators, their frequency, and dependency. The evaluation result confirms applicability of the metrics, and the practical guidelines about the mutation operators are concluded from the experimental data.

Keywords Mutation testing · Mutant clustering · Mutation operators · C#

1 Introduction

Mutation testing is a method for evaluating a quality of a test case suite and/or creating a set of test cases [1]. The main idea originates from the fault injection techniques. A change is introduced to a program under test. The change represents typically a possible mistake made by a programmer. It is assumed that the change could not be revealed by a simple program compilation. A changed program is called a *mutant*.

The main obstacle in application of a mutation testing process is its high cost. Therefore there are many approaches to its cost reduction [1, 2]. Some of them

A. Derezińska (✉)
Institute of Computer Science, Warsaw University of Technology,
Nowowiejska 15/19, 00-665 Warsaw, Poland
e-mail: A.Derezinska@ii.pw.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_33

are based on lowering of the number of generated mutants and/or the number of performed test cases. Mutant clustering is one of the cost reduction methods that was considered for the mutation testing process [3, 4].

The first results on mutant clustering for object-oriented programs were reported in [5]. A general experimental scenario was proposed for evaluation of the tradeoff between mutation score accuracy and the complexity of a mutation testing process expressed in a number of generated mutants and a number of test cases. The scenario was adapted to three cost reduction techniques: selection of mutants, mutant sampling, and clustering. The detailed results of mutant clustering experiments for C# programs, the experimental scenario, and evaluation of a quality metric are given in [6].

This paper addresses another problem of mutant clustering. It is a question of how we can generalize the results of experiments on mutant clustering, which might be useful for other projects. Especially, it is interesting how to evaluate relations of mutants generated by different mutation operators. Therefore, three new metrics were developed, dealing with usefulness of mutants generated by a given operator, their frequency, and dependency among mutants. Based on these metrics and data gathered in our previous experiments [5, 6] the approach was applied. We could examine in a quantitative way the differences between standard and object-oriented operators, distinguish a pair of complementary operators, and classify operators that cannot be omitted in order to preserve the mutation result accuracy.

The paper is organized as follows: Sect. 2 describes briefly the basic notion of mutation testing and mutant clustering method. In Sect. 3 metrics used for analysis of mutant clustering results are introduced and illustrated by an example. Section 4 presents an experiment overview and results of the conducted experiments. Finally, Sect. 5 concludes the work.

2 Background

In this section basic concepts of mutation testing as well as an idea of mutant clustering and related works related to it are discussed.

2.1 Mutation Testing

In mutation testing, a program change is specified by a *mutation program operator* and introduced in an automatic way using a mutation tool. *Standard*, or so-called traditional, mutation operators deal with the common programming features, typical to all programming languages, like arithmetic, logical and relational operators, assignment statements, constant usage, etc. Different specialized programming features are also covered by the devoted mutation operators. Features characteristic to object-oriented languages (OO in short) are handled by *object-oriented mutation* *operators* proposed, for example, for Java [7] or C# [8, 9]. If a mutation operator is applied only once and in one place of a program, we speak about *first order mutation*.

Evaluation of a test suite is performed in a mutation testing process. For a given program and a set of selected mutation operators, a set of mutants is created. The mutants are run against tests from a test suite under concern. If a mutant behavior is different from the behavior of the original program, the mutant is said to be *killed*. Tests that are able to kill mutants should be good at revealing mistakes represented by the mutation operators. A mutation testing result, called a *mutation score* (MS), is calculated as a ratio of the number of all killed mutants over the number of all nonequivalent mutants. A mutant is *equivalent* if its behavior cannot be distinguished from the original program by any test. In many practical cases, instead of an exact mutation score, its approximate value is calculated, because it is not possible to classify exactly all equivalent mutants in an automatic way.

2.2 Mutant Clustering

There are many approaches to reduction of mutation testing costs based on lowering of considered mutants and therefore reducing also the number of test runs [1, 2]. One such analyzed solution was mutant clustering [3, 4].

The main idea of the mutant clustering originates on the concept of equivalence partitioning. A set of all mutants of a program is divided into groups, called clusters. The division is realized in the context of a given test set, similarly as in mutation score evaluation. Each group is characterized by the similar ability of being killed by the same subset of tests. Allocation of a mutant to a group can be realized by a clustering algorithm such as agglomerative hierarchical or K-means clustering [3, 10].

The mutant clustering is specified for a given set of mutants of a program under test, and a given set of tests. A threshold K denotes a resemblance between mutant groups. Two groups are said to be *similar with K degree*, if the number of tests that kill at least one mutant from one group and kill none mutant from the former group equals K.

Next, in the mutation testing process, instead of all mutants, only one mutant for each group is used. This mutant represents the group (cluster) that should have the comparable features, as far as the subset of tests associated with this group is concerned. Usage of a reduced number of mutants lowers the mutation costs. However, the accuracy of the mutation score can be declined.

2.3 Related Works

Primary experiments on mutant clustering in mutation testing were conducted for C programs [3]. They reported considerable potential benefits, for example, usage of 13 % of all mutants and 8 % of tests gave a mutation score of a high accuracy (99 %). However, this result referred to a simple, not object-oriented programming language and only standard mutation operators, which usually are more redundant.

The above-mentioned result was based on full data, i.e., all mutants run against all tests. The practical solution to mutation clustering based on a static domain analysis was presented in [4]. The proof of concept was illustrated by a small Java program, for which satisfactory results were obtained, namely after running 25 % of mutants with 62 % of tests the mutation score was equal to 94 % of the exact mutation score.

Mutant clustering in the context of object-oriented operators was studied for the first time in the experimental process for comparing of different cost reduction techniques [5]. The detailed analysis of mutant clustering was discussed in [6]. The quantitative data of this experiment is recalled in Sect. 4.1. This paper provides further methods of clustering data analysis in order to generalize the results to other projects.

The research on cost reduction methods applied to C# programs was performed only by the author of [5, 6]. Most of the other work on object-oriented programs was done for Java programs [11–13], but the clustering method was not considered.

3 Metrics for Generalization of Clustering Results

Cluster of mutants includes mutants generated by various mutation operators. Many mutants created with the same mutation operator (*Op*) can contribute to the same cluster $\{Op1, Op2, \ldots\}$. Therefore, there can exist clusters that are mainly constituted by mutants of selected mutation operators. These mutants are the most probable representative of these clusters.

Mutants of the same operator can also be met in many clusters. Some pairs of operators can be associated and encounter in the same clusters. In such case it could be possible to omit one of the operators.

In order to quantitatively evaluate such phenomena the following metrics were used.

3.1 Metric Definitions

Three additional metrics were proposed in order to evaluate and compare the clustering results. Each metric is calculated for a mutation operator (*Op*).

The first metric is *usefulness of mutants* (*UM*). It calculates how big a subset of mutants is that are useful in the context of a given operator.

$$UM(Op) = \frac{NG(Op)}{NM(Op)} \tag{1}$$

where

$NG(Op)$—the number of groups, in which at least one mutant exists that was created using the *Op* mutation operator,

$NM(Op)$—the number of all mutants generated using the *Op* mutation operator.

The second metric, so-called *frequency* (*FR*), examines frequency of an operator occurrence. It calculates the amount of groups, which includes at least one mutant designed by the operator in relation to all group number.

$$FR(Op) = \frac{NG(Op)}{NG_{All}} \tag{2}$$

where $NG(Op)$—as above, and NG_{All}—number of all groups.

The third metric is called *dependency* (*DEP*). It evaluates dependency of an ordered pair of mutation operators.

$$DEP(Op_1, Op_2) = \frac{NPM(Op_1, Op_2)}{NM(Op_1)} \tag{3}$$

where

$NM(Op_1)$—a number of all mutants generated using the Op_1 mutation operator.

$NPM(Op_1, Op_2)$—a number of occurrences of mutant pairs created with Op_1 and Op_2 operators. This value is calculated as a sum of all other groups of a minimum of two numbers: a number of mutants of a given group created using Op_1, and an analogous number of mutants created by the second operator Op_2.

$$NPM(Op_1, Op_2) = \sum_{g \in G} \min(NM(g, Op_1), NM(g, Op_2)) \tag{4}$$

where

$NM(g, Op_1)$—a number of mutants from the group g that were generated using the Op_1 mutation operator.

It should be noticed, that the operator dependency metric is not symmetric, i.e. $DEP(Op_1, Op_2) \neq DEP(Op_2, Op_1)$.

3.2 Example

The metrics will be illustrated with a simple example. Three mutation operators were used for generation of mutants: EOC, IOP, and EXS. (The full operator names are listed in Table 1). Using EOC operator five mutants were created. Four mutants were generated with IOP operator and one mutant with EXS.

After performing an algorithm of mutant clustering four groups of mutants were specified. The result groups consist of the following mutants:

$G1 = \{EOC1, EOC2, IOP1, IOP2\}$
$G2 = \{EOC4, EOC5, IOP4\}$
$G3 = \{EOC3\}$
$G4 = \{IOP3, EXS1\}$

Table 1 Standard and object-oriented mutation operators (C# supported by CREAM v.3)

No	Operator type	Abbr.	Name of mutation operator
1	Standard	ABS	Absolute value insertion
2	Standard	AOR	Arithmetic operator replacement $(+, -, *, /, \%)$
3	Standard	ASR	Assignment operator replacement $(=, +=, -=, /=, *=)$
4	Standard	LCR	Logical connector replacement $(\&\&, \vert\vert)$
5	Standard	LOR	Logical operator replacement $(\&, \vert,{}^\wedge)$
6	Standard	ROR	Relational operator replacement $(<, <=, >, >=, ==, !=)$
7	Standard	UOI	Unary operator insertion $(+, -, !, \sim)$
8	Standard	UOR	Unary operator replacement $(++, --)$
1	Object-oriented	DMC	Delegated method change
2	Object-oriented	EHR	Exception handler removal
3	Object-oriented	EOA	Reference assignment and content assignment replacement
4	Object-oriented	EOC	Reference comparison and content comparison replacement
5	Object-oriented	EXS	Exception swallowing
6	Object-oriented	IHD	Hiding variable deletion
7	Object-oriented	IHI	Hiding variable insertion
8	Object-oriented	IOD	Overriding method deletion
9	Object-oriented	IOK	Overriding method substitution
10	Object-oriented	IOP	Overriding method calling position change
11	Object-oriented	IPC	Explicit call of a parent's constructor deletion
12	Object-oriented	ISK	Base keyword deletion
13	Object-oriented	JID	Ember variable initialization deletion
14	Object-oriented	JTD	This keyword deletion
15	Object-oriented	OAO	Argument order change
16	Object-oriented	OMR	Overloading method contents change
17	Object-oriented	PRM	Property replacement with member field
18	Object-oriented	PRV	Reference assignment with other compatible type

The usefulness metric *UM* was calculated for each operator in the following way:

$$\begin{aligned} UM(EOC) &= \frac{NG(EOC)}{NM(EOC)} = \frac{3}{5} = 0.6 \\ UM(IOP) &= \frac{NG(IOP)}{NM(IOP)} = \frac{3}{4} = 0.75 \\ UM(EXS) &= \frac{NG(EXS)}{NM(EXS)} = \frac{1}{1} = 1.0 \end{aligned} \tag{5}$$

The calculated values can be interpreted as a useful part of mutants. For example, 60 % of mutants could be selected for the EOC operator and still in each group there would be at least one mutant created by this operator. However, all mutants

(100 %) generated by the EXS operator are indispensable in order to ensure the same condition.

The frequency metric calculated for the example mutants gives the following values:

$$
\begin{aligned}
FR(EOC) &= \frac{NG(EOC)}{NG_{All}} = \frac{3}{4} = 0.75\\
FR(IOP) &= \frac{NG(IOP)}{NG_{All}} = \frac{3}{4} = 0.75\\
FR(EXS) &= \frac{NG(EXS)}{NG_{All}} = \frac{1}{4} = 0.25
\end{aligned}
\qquad (6)
$$

For a given operator, the metric assesses the frequency of mutants belonging to groups. For example, the metric of EOC is equal to 0.75. It means that 75 % of all groups include at least one mutant created using this operator.

Finally, the dependency metric is calculated for any ordered pair of mutation operators. We choose for example two operators EOC and IOP. Because the metric is not symmetric, two ordered pairs are considered: (EOC, IOP) and (IOP, EOC).

$$
\begin{aligned}
DEP(EOC, IOP) &= \frac{NPM(EOC, IOP)}{NM(EOC)} = \frac{3}{5} = 0.6 \qquad (7)\\
DEP(IOP, EOC) &= \frac{NPM(IOP, EOC)}{NM(IOP)} = \frac{3}{4} = 0.75
\end{aligned}
$$

Based on the first value (0.6) we can deduce that 60 % of mutants created by EOC can be substituted by IOP mutants. In the opposite case the value is different and is equal to 0.75. This means that 75 % of IOP mutants have a pair of EOC mutants in a group. Comparing both values of the metric, we can conclude that in this example it is better to substitute operator IOP by EOC (0.75) than vice versa (0.6).

4 Experiments on Mutation Clustering

Evaluation of the approach will be presented on experimental data. The metrics were applied to the analysis of mutation clustering results gathered in the experiments on standard and object-oriented mutation of C# programs [5, 6].

4.1 Experiment Setup

Data for the mutant clustering and evaluation of the metrics were collected in experiments carried out with the CREAM v3 tool. CREAM is a mutation testing tool for C# programs [14]. It was the first tool that supported object-oriented mutation operators for C# programs [15, 16]. Its third version was enhanced with an extension for efficient performing and evaluation experiments on cost reduction techniques: selection of mutants, mutant sampling, and clustering [5]. The tool supports 18 object-oriented operators and eight standard ones (Table 1).

The experiments were conducted on three commonly used open-source programs, Enterprise Logging (http://entlib.codeplex.com), Castle (http://www.castleproject.org) and Mono-Gendarme (http://www.mono-project.com/Gendarme). All first order mutants were generated for the mutation operators given in Table 1. Additionally, only mutants covered by tests from a given test suite were considered, as not covered mutants were not able to be killed by tests. Then all mutants were run against all test cases. The collected results were stored and used in the evaluation process of the cost reduction techniques [5]. For different cost reduction method the appropriate quality measures were calculated that allow to express the tradeoff between mutation score and the number of mutants and the number of tests.

The detailed results of the basic quality analysis of the mutation clustering approach are presented in [6]. For all mutants the agglomerative clustering algorithm was applied. Mutants generated by standard mutation operators and by object-oriented ones (in short—standard and OO mutants) were analyzed separately. The groups of mutants were formulated for the K degree of the clustering algorithm varying from 0 to 19. According to the quality analysis the best results were obtained, for $K = 1$ in case of object-oriented operators and $K = 2$ in case of the standard operators, assuming that the mutation score adequacy contributes of 60 % to the overall quality, whereas number of mutants and number of tests of 20 % each. The experiments showed that it was possible to use 32 % of OO mutants and 18 % of tests to obtain the mutation score of 97 % close to the original one (i.e. calculated using all OO mutants and all test). The analogues data for the best results of standard mutation was 19 % of mutants, 22 % of tests and 91 % of mutation score accuracy.

4.2 Evaluation of Mutation Clustering Results

Mutation data from the above-mentioned experiments were used in the further evaluation of mutation clustering results addressing the generalization problem. The evaluation was based on the metrics specified in Sect. 3. The results were analyzed separately for standard and object-oriented mutations. The metrics were calculated in respect to all mutation operators used in experiments.

Results of two metrics, *usefulness of mutants* and *frequency*, calculated for the subject programs and their average values are shown in Table 2. The upper part of the table includes values of standard operators, whereas the lower part gives data for OO operators. Empty places, denoted by '–' character, correspond to cases when no mutant was generated from a given program (column) using this kind of operator (row).

Analyzing the first metric for object-oriented operators, we can observe that in most of the cases the value of usefulness of mutants is relatively high. A value

Table 2 Usefulness of mutants (UM) and frequency (FR) metrics for standard and object-oriented mutation operators

Oper.	Usefulness of mutants metric (*UM*)				Frequency metric (*FR*)			
	Logging	Castle	Gendarme	Average	Logging	Castle	Gendarme	Average
ABS	0.57	0.89	1.00	0.82	0.01	0.01	0.00	0.01
AOR	0.11	0.42	0.33	0.29	0.10	0.01	0.02	0.04
ASR	0.42	0.53	0.56	0.50	0.12	0.03	0.03	0.06
LCR	0.93	0.82	0.73	0.83	0.07	0.17	0.18	0.14
LOR	–	–	0.64	0.64	–	–	0.01	0.01
ROR	0.54	0.33	0.33	0.40	0.22	0.21	0.18	0.20
UOI	0.50	0.61	0.46	0.52	0.76	0.75	0.81	0.77
UOR	0.40	0.42	0.42	0.41	0.02	0.08	0.04	0.05
DMC	–	–	–	–	–	–	–	–
EHR	1.00	1.00	0.60	0.87	0.02	0.01	0.01	0.01
EOA	–	1.00	1.00	1.00	–	0.01	0.01	0.01
EOC	0.98	0.62	0.65	0.75	0.14	0.34	0.37	0.28
EXS	1.00	1.00	–	1.00	0.00	0.01	–	0.01
IHD	–	–	–	–	–	–	–	–
IHI	–	–	–	–	–	–	–	–
IOD	1.00	1.00	0.78	0.93	0.07	0.03	0.02	0.04
IOK	1.00	1.00	0.75	0.92	0.06	0.02	0.02	0.04
IOP	0.43	1.00	1.00	0.81	0.01	0.01	0.08	0.03
IPC	0.97	0.45	–	0.71	0.12	0.04	–	0.08
ISK	0.81	0.64	1.00	0.82	0.09	0.02	0.11	0.07
JID	0.66	0.67	0.65	0.66	0.07	0.18	0.31	0.19
JTD	0.92	1.00	–	0.96	0.16	0.10	–	0.13
OAO	0.46	0.63	0.48	0.53	0.18	0.19	0.11	0.16
OMR	0.56	0.84	–	0.70	0.03	0.11	–	0.07
PRM	0.64	0.83	0.90	0.79	0.02	0.03	0.03	0.03
PRV	0.24	0.37	0.34	0.32	0.14	0.09	0.04	0.09

can be counted as high if it is bigger than 0.8 for at least one program. Eight OO operators have at least one 1.0 (100 %) value, which means that for this program all mutants generated by this operator contribute as group representatives and could not be omitted in the mutation score analysis.

However, we can observe the PRV operator (*Reference Assignment with other Compatible Type*) for which this metric is low, i.e., about 0.3 for each program. Generating only about 32 % of all PRV mutants it is possible to create the same groups considering their member operators. Moreover, analyzing the frequency metric for PRV we have found that in average 9 % of groups includes at least one PRV mutant. This result is medium high in comparison to other operators but not negligible.

In conclusion, it is worthwhile to limit the number of PRV mutants, as it is possible to reduce the mutant number considerably without loss of the mutation score accuracy.

Comparing results of the usefulness metric for object-oriented and standard operators we can observe that in general the values of standard operators are lower than the object-oriented ones. Only two standard operators (ABS and LCR) have a high value of the first metric. This confirms the other results [5, 11, 17] that among standard mutants can be more surplus (redundant) mutants than in the object-oriented mutants. It should be noted that this effect is visible although the set of standard operators of CREAM and therefore used in this experiment was very limited. It was based mainly on the operators classified as selective in the standard operator analysis [17].

Analogues reasoning for the PRV operator can be performed for selected standard operates, in particular ROR and UOR. In case of these operators, according to the first metric at least 40 % of mutants should be generated for each operator.

Results of the third metric—*dependency* are shown in Table 3 for standard mutation operators and in Table 4 for object-oriented ones. The tables include values averaged over all three programs examined in experiments. Operators DMC, IHD were omitted as no mutants were generated by them in the considered programs.

Analyzing the object-oriented operators, we can see that the maximum values 0.87 and 0.79 are calculated for two operators IOK and IOD. This result denotes that most mutants generated using the IOK operator (87 %) are in the same group as mutants created by IOD operator. The opposite dependency is satisfied in 79 %. Therefore, we can assume that resigning one such operator can reduce the mutation testing cost without considerable loss of the mutation score accuracy, because they are complementary, i.e., mutants of one operator can be substituted by mutants of the second operator. The slightly better choice is selection of IOK, because DEP(IOK, IOD) > DEP(IOD, IOK).

The dependency metric calculated for other pairs of object-oriented operators give in most cases very low results (about few %) or for several pairs results about 10–20 %. Therefore we cannot point at any other pair of object-oriented operators as being dependent in general.

Considering the third metric for standard mutation operators (Table 3), we can find more results above 50 % than for the object-oriented operators. Five standard

Table 3 Dependency metric for standard operators

	ABS	AOR	ASR	LCR	LOR	ROR	UOI	UOR	Sum
ABS	–	0.11	0.19	0.15	0.33	0.48	0.75	0.19	2.2
AOR	0.00	–	0.06	0.04	0.03	0.33	0.51	0.22	1.19
ASR	0.02	0.10	–	0.07	0.04	0.25	0.60	0.09	1.17
LCR	0.00	0.02	0.04	–	0.00	0.15	0.37	0.05	0.63
LOR	0.07	0.14	0.14	0.07	–	0.64	0.57	0.36	1.99
ROR	0.00	0.05	0.04	0.05	0.02	–	0.49	0.10	0.75
UOI	0.01	0.09	0.05	0.05	0.00	0.17	–	0.04	0.41
UOR	0.01	0.12	0.07	0.08	0.05	0.44	0.53	–	1.3

Table 4 Dependency metric for object-oriented operators

	EHR	EOA	EOC	EXS	IHI	IOD	IOK	IOP	IPC	ISK	JID	JTD	OAO	OMR	PRM	PRV	SUM
EHR	–	0	0	0	–	0	0	0	0	0	0.11	0	0.22	0	0	0	0.33
EOA	0	–	0.25	0	–	0	0	0	0	0	0	0	0	0	0	0	0.25
EOC	0	0	–	0	–	0.02	0.01	0	0	0	0.05	0.04	0.02	0.01	0	0.01	0.16
EXS	0	0	0.17	–	–	0	0	0	0	0	0	0	0	0	0	0	0.17
IHI	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	0
IOD	0	0	0.16	0	–	–	0.79	0	0.03	0	0.14	0.05	0.11	0.05	0	0.02	1.35
IOK	0	0	0.14	0	–	0.87	–	0	0.03	0	0.11	0.06	0.10	0.06	0	0.02	1.39
IOP	0	0	0	0	–	0	0	–	0	0.26	0.21	0	0.26	0	0.02	0	0.75
IPC	0	0	0.02	0	–	0.01	0.01	0	–	0	0	0.02	0.02	0.02	0	0	0.1
ISK	0	0	0.03	0	–	0	0	0.05	0	–	0.06	0	0.04	0.02	0	0	0.2
JID	0	0	0.07	0	–	0.03	0.03	0.01	0	0.01	–	0.04	0.01	0.02	0	0.04	0.26
JTD	0	0	0.10	0	–	0.01	0.01	0	0.01	0	0.08	–	0.06	0.03	0.01	0	0.31
OAO	0.01	0	0.04	0	–	0.01	0.01	0.01	0	0.01	0.02	0.02	–	0	0	0.02	0.15
OMR	0	0	0.06	0	–	0.01	0.01	0	0.01	0.03	0.04	0.02	0	–	0	0.01	0.19
PRM	0	0	0.03	0	–	0	0	0.03	0	0	0.03	0.04	0.03	0	–	0.19	0.35
PRV	0	0	0.03	0	–	0	0	0	0	0	0.02	0	0.03	0.01	0.03	–	0.12

operators, namely ABS, AOR, ASR, LOR, and UOR can be partially substituted by the UOI operator. On the other hand the dependency is in the range of 50–60 % (only for ABS equal 75 %). Therefore the dependency is not as definite as in the case of the pair of IOD-IOK operators.

The last column in the tables with dependency metric includes the sum of the numbers in the corresponding row. This sum represents information about to what extent mutants created by the row operator can be substituted by any combination of the remaining operators. The lower the sum, the more reasonable it is to retain this operator in the mutation testing process.

Analyzing this sum of object-oriented operators, we can see that the highest values are for operators IOD, IOK (already recognized as complementary ones) and the IOP operator. Therefore the remaining object-oriented operators should be applied, i.e., the majority of OO operators. This evaluation confirms the fact that OO operators correspond to various advanced programming features and need more specific tests.

For the standard operators the sums are in general higher than for OO operators. Operators LCR, ROR, and UOI turned out to be the most applicable as they have the sum below zero. This result is consistent with the findings on the selective mutation in C# programs [5].

5 Conclusions

This paper presents a study on the result evaluation of mutation clustering. It copes with the question: How can the clustering results be generalized and associated with the selection of mutation operators. The problem was examined with three new metrics about mutant usefulness, frequency, and dependency in terms of mutation operators used for the mutant generation. The metrics were applied to mutant clustering results on three real-world C# programs mutated with standard and object-oriented operators.

Combining the results of usefulness and frequency metrics, we can observe that reducing the number of generated PRV mutants gives noticeable mutant cost reduction without a loss of the mutation score accuracy. It is also worthwhile to select only one operator among IOK and IOD operators. The lessons learned point at different characteristics between structural and object-oriented mutation operators. In general, less OO mutation operators can be omitted if an adequate mutation result has to be assured. This fact can be caused by the higher specialization of the OO mutation operators than the standard ones. For the standard mutation operators, even basing of a preliminary reduced set of operators (including all selective according to [17]), we can still reduce the number of generated mutants. Among standard operators the most useful were LCR, ROR, and UOI, which corresponds to the results on selective mutation in OO programs based on other methodologies [5, 11].

Acknowledgments I am thankful to M. Rudnik for his contribution to the CREAM tool development and for performing mutation testing experiments.

References

1. Jia, Y., Harman, M.: An analysis and survey of the development of mutation testing. IEEE Trans. Softw. Eng. **37**(5), 649–678 (2011)
2. Usaola, M.P., Mateo, P.R.: Mutation testing cost reduction techniques: a survey. IEEE Softw. **27**(3), 80–86 (2010)
3. Hussain, S.: Mutation Clustering. Master's Thesis. King's College London, Strand, London (2008)
4. Ji, C., Chen, Z.Y., Xu, B.W., Zhao, Z.H.: A novel method of mutation clustering based on domain analysis. In: Proceedings of 21st International Conference on Software Engineering & Knowledge Engineering 422–425 (2009)
5. Derezińska, A., Rudnik, M.: Quality evaluation of object-oriented and standard mutation operators applied to C# programs. In: Furia, C.A., Nanz, S. (eds.) TOOLS Europe. LNCS, vol. 7304, pp. 42–57. Springer, Berlin (2012)
6. Derezińska, A.: A quality estimation of mutation clustering in C# programs. In: Zamojski, W. (ed.) New Results in Dependability & Computer Systems. AISC, vol. 224, pp. 119–129. Springer, Switzerland (2013)
7. Ma, Y.-S., Kwon, Y.-R., Offut, A.J.: Inter-class mutation operators for Java. In: Proceedings of 13-th International Symposium on Software Reliability Engineering, pp. 352–363. IEEE Computer Society (2002)
8. Derezińska, A.: Advanced mutation operators applicable in C# programs. In: Sacha, K. (ed.) Software Engineering Techniques: Design for Quality. IFIP, vol. 227, pp. 283–288. Springer, Boston (2006)
9. Derezińska, A.: Quality Assessment of Mutation Operators Dedicated for C# Programs. In: 6th International Conference on Quality Software, QSIC'06, Beijing, China, pp. 227–234, IEEE Computer Society Press, California (2006)
10. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Comput. Surv. **31**(3), 264–323 (1999)
11. Hu, J., Li, N., Offutt, J.: An analysis of OO mutation operators. In: Proceedings of 4th International Conference Software Testing Verification and Validation Workshops, pp. 334–341 (2011)
12. Zhang, L., Gligoric, M., Marinov, D., Khurshid, S.: Operator-based and random mutant selection: better together. In: 28th IEEE/ACM Conference on Automated Software Engineering (ASE 2013), pp. 92–102. Palo Alto (2013)
13. Bluemke, I., Kulesza, K.: Reduction in mutation testing of Java classes. In: Proceedings of International Joint Conference on Software Technologies (ICSOFT). Vienna (2014)
14. CREAM, http://galera.ii.pw.edu.pl/~adr/CREAM/
15. Derezińska, A., Szustek, A.: Tool-supported mutation approach for verification of C# programs. In: Zamojski, W., et al. (eds.) Proceedings of International Conference on Dependability of Computer Systems, DepCoS-RELCOMEX'08, pp. 261–268 (2008)
16. Derezińska, A., Szustek, A.: Object-oriented testing capabilities and performance evaluation of the C# mutation system, In: Szmuc, T., Szpyrka, M., Zendulka, J. (eds.) CEE-SET 2009. LNCS, vol. 7054, pp. 229–242 (2012)
17. Offut, J., Rothermel, G., Zapf, C.: An experimental evaluation of selective mutation. In: Proceedings of 15th International Conference on Software Engineering, pp. 100–107 (1993)

TRACO Parallelizing Compiler

Marek Palkowski and Wlodzimierz Bielecki

Abstract This paper presents a source-to-source compiler, TRACO, for automatic extraction of both coarse- and fine-grained parallelism available in C/C++ loops. Parallelization techniques, implemented in TRACO, are based on the transitive closure of a relation describing all the dependences in a loop. Coarse- and fine-grained parallelism is represented with synchronization-free slices (space partitions) and a legal loop statement instance schedule (time partitions), respectively. On its output, TRACO produces compilable parallel OpenMP C/C++ and/or OpenACC C/C++ code. The effectiveness of TRACO and efficiency of parallel code produced by TRACO are evaluated by means of the NAS Parallel Benchmark and Polyhedral Benchmark suites.

Keywords Source-to-source parallelizing compiler · Fine- and coarse-grained parallelism · Free scheduling · Transitive closure

1 Introduction

Parallel computer programs are more difficult to write than sequential ones. Exposing parallelism in serial programs and writing parallel programs without applying parallelizing compilers decrease the productivity of programmers and increase the time and cost of producing parallel programs. Because for many applications, most computations are contained in program loops, automatic extraction of parallelism available in loops is extremely important for multicore systems.

M. Palkowski (✉) · W. Bielecki
Faculty of Computer Science and Information Systems, West Pomeranian University of Technology, Zolnierska 49, 71210 Szczecin, Poland
e-mail: mpalkowski@wi.zut.edu.pl
URL: http://www.wi.zut.edu.pl

W. Bielecki
e-mail: wbielecki@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_34

The goal of this paper is to present an open source parallelizing compiler, TRACO, implementing loop parallelization approaches based on transitive closure.

The input of TRACO is a C program, while the output is an OpenMP C/C++ or OpenACC C/C++ program. TRACO extracts both coarse- and fine-grained parallelism. It also uses variable privatization and parallel reduction techniques to reduce the number of dependence relations; this leads to reducing parallelization time and extending the scope of the TRACO applicability. The compiler includes a preprocessor of the C program, data dependence analyzer, parallelization engine, code generator, and postprocessor. To the best of our knowledge, there are no source-to-source compilers based on the transitive closure of dependence relation graphs.

Results of a comparative analysis of TRACO features and those demonstrated by Pluto, Par4all, Cetus, and ICC have been discussed.

2 Background

In this paper, we deal with affine loop nests where, for given loop indices, lower and upper bounds as well as array subscripts and conditionals are affine functions of surrounding loop indices and possibly of structure parameters (defining loop indices bounds), and the loop steps are known constants.

Algorithms implemented in TRACO require an exact representation of loop-carried dependences and consequently an exact dependence analysis which detects a dependence if and only if it actually exists. To describe and implement parallelization algorithms, we chose the dependence analysis proposed by Pugh and Wonnacott [18], where dependences are represented with dependence relations.

A dependence relation is a tuple relation of the form [*input list*]→[*output list*]: *formula*, where *input list* and *output list* are the lists of variables and/or expressions used to describe input and output tuples and *formula* describes the constraints imposed upon *input list* and *output list* and it is a Presburger formula built of constraints represented with algebraic expressions and using logical and existential operators [18].

Standard operations on relations and sets are used, such as intersection (∩), union (∪), difference (−), domain (dom R), range (ran R), and relation application ($S' = R(S)$: $e' \in S'$ iff exists e s.t. $e \rightarrow e' \in R$, $e \in S$). In detail, the description of these operations is presented in [11, 18].

The positive transitive closure for a given relation R, R^+, is defined as follows [11]:

$$R^+ = \{e \rightarrow e' : e \rightarrow e' \in R \vee \exists e'' s.t.\ e \rightarrow e'' \in R \wedge e'' \rightarrow e' \in R^+\}. \quad (1)$$

It describes which vertices e' in a dependence graph (represented by relation R) are connected directly or transitively with vertex e.

Transitive closure, R^*, is defined as follows [12]: $R^* = R^+ \cup I$, where I is the identity relation. It describes the same connections in a dependence graph (represented by R) that R^+ does plus connections of each vertex with itself.

To facilitate the exposition and implementation of TRACO algorithms, we have to preprocess dependence relations making their input and output tuples to be of the same dimension and to contain the identifiers of statements responsible for the source and destination of each dependence. The preprocessing algorithm is presented in paper [3].

Given a relation R, found as the union of all (preprocessed) dependence relations extracted for a loop, the iteration space, S_{DEP}, including dependent statement instances is formed as domain(R) ∪ range(R). A set, S_{IND}, comprising independent statement instances is calculated as the difference between the set of all statement instances, S_{SI}, and the set of all dependent statement instances, S_{DEP}, i.e., $S_{IND} = S_{SI} - S_{DEP}$. To scan elements of sets S_{DEP} and S_{IND} in the lexicographic order, we can apply any well-known code generation technique [2, 11].

3 Coarse-Grained Parallelism Extraction Using Iteration Space Slicing

Algorithms presented in paper [3] are based on transitive closure and allow us to generate parallel code representing synchronization-free slices or slices requiring occasional synchronization.

Definition 1 Given a dependence graph defined by a set of dependence relations, a slice S is a weakly connected component of this graph.

Definition 2 An ultimate dependence source is a source that is not the destination of another dependence. Ultimate dependence sources represented by relation R can be found by means of the following calculations: domain(R)—range(R). The set of ultimate dependence sources of a slice forms the set of its sources.

Definition 3 The representative source of a slice is its lexicographically minimal source.

An approach to extract synchronization-free slices implemented in TRACO takes two steps [3]. First, for each slice, a representative statement instance is defined (it is the lexicographically minimal statement instance from all the sources of a slice). Next, slices are reconstructed from their representatives and code scanning these slices is generated.

Given a dependence relation R describing all the dependences in a loop, a set of statement instances, S_{UDS}, are calculated. It describes all ultimate dependence sources of slices as

$$S_{UDS} = domain(R) - range(R). \qquad (2)$$

In order to find elements of S_{UDS} that are representatives of slices, we build a relation, R_{USC}, that describes all pairs of the ultimate dependence sources being transitively connected in a slice, as follows:

$$R_{USC} = \{[e] \rightarrow [e'] : e, e' \in S_{UDS}, e \prec e', (R^*(e) \cap R^*(e'))\}. \tag{3}$$

The condition ($e \prec e'$) in the constraints of relation R_{USC} above means that e is lexicographically smaller than e'. The intersection $(R^*(e) \cap R^*(e'))$ in the constraints of R_{USC} guarantees that vertices e and $e'$are transitively connected, i.e., they are the sources of the same slice.

Next, set S_{repr} containing representatives of each slice is found as $S_{repr} = S_{UDS}$—range(R_{USC}). Each element e of set S_{repr} is the lexicographically minimal statement instance of a synchronization-free slice. If e is the representative of a slice with multiple sources, then the remaining sources of this slice can be found applying relation $(R_{USC})^*$ to e, i.e., $(R_{USC})^*(e)$. If a slice has the only source, then $(R_{USC})^*(e) = e$. The elements of a slice represented with e can be found applying relation R^* to the set of sources of this slice:

$$S_{slice} = R^*((R_{USC})^*(e)). \tag{4}$$

Any tool to generate code for scanning polyhedra can be applied to produce parallel pseudocode, for example, the CLOOG library [2] or the *codegen* function of the Omega project [11].

4 Variable Privatization and Parallel Reduction

TRACO automatically recognizes loop variables that can be safely privatized and/or can be used for parallel reduction. Applying this technique permits us to reduce the number of dependence relations.

Privatization is a technique that allows each concurrent thread to allocate a variable in its private storage such that each thread accesses a distinct instance of a variable.

Definition 4 A scalar variable x defined within a loop is said to be privatizable with respect to that loop if and only if every path from the beginning of the loop body to a use of x within that body must pass through a definition of x before reaching that use [13].

Definition 5 Given n inputs $x_1, x_2, \ldots, x_n$ and an associative operation $\otimes$, a parallel *reduction* algorithm computes the output $x_1 \otimes x_2 \otimes \cdots \otimes x_n$ [16].

The idea of recognizing variables to be privatized and/or used for parallel reduction and being implemented in TRACO is the following. The first step is to search for scalar or one-dimensional array variables for privatization. A variable can be privatized if the lexicographically first statement in the loop body referring to this variable does not read its value, i.e., the first access to this variable is a write operation [13].

Next, we seek for variables that are involved in reduction dependences only (they cannot be involved in other types of dependences). Then we check whether there exist dependence relations refereeing to variables which cannot be privatized or used for parallel reduction. If no, this means that privatization and parallel reduction eliminate all the dependences in the loop, thus its parallelization is trivial. Otherwise, we form a set including: (i) dependence relations not being eliminated by means of variable privatization and reduction and (ii) dependence relations describing dependences not carried by loops and referring to variables to be privatized. Finally, we generate output code using the set mentioned above and a set including variables to be used for parallel reduction.

5 Finding (Free) Scheduling for Parameterized Loops

The algorithm, presented in our paper [4], allows us to generate fine-grained parallel code based on the free schedule representing time partitions; all statement instances of a time partition can be executed in parallel, while partitions are enumerated sequentially. The free schedule function is defined as follows:

Definition 6 ([8]) The *free schedule* is the function that assigns discrete time of execution to each loop statement instance as soon as its operands are available, that is, it is mapping σ:*LD*$\rightarrow$ $\mathbb{Z}$ such that

$$\sigma(p) = \begin{cases} 0\, if\ there\ is\ no\ p_1 \in LD\ s.t.\ p_1 \rightarrow p \\ 1 + max(\sigma(p_1), \sigma(p_2), \ldots, \sigma(p_n));\ p, p_1, p_2, \ldots, p_n \in LD; \\ p_1 \rightarrow p, p_2 \rightarrow p, \ldots, p_n \rightarrow p, \end{cases} \quad (5)$$

where $p, p_1, p_2, \ldots, p_n$ are loop statement instances, *LD* is the loop domain, $p_1 \rightarrow p, p_2 \rightarrow p, \ldots, p_n \rightarrow p$ mean that the pairs p_1 and p, p_2 and $p, \ldots, p_n$ and p are dependent, p represents the destination while $p_1, p_2, \ldots, p_n$ represent the sources of dependences, n is the number of operands of statement instance p (the number of dependences whose destination is statement instance p).

The free schedule is the fastest legal schedule [8].

The idea of the algorithm to extract time partitions applying transitive closure is as follows [4]. Given preprocessed relations $R_1, R_2, \ldots, R_m$, we first calculate $R = \bigcup_{i=1}^{m} R_i$. Next, we create a relation R' by inserting variables k and $k+1$ into the first position of the input and output tuples of relation R; variable k is to present the time of a partition (a set of statement instances to be executed at time k). Next, we calculate the transitive closure of relation R', $R'*$, and form the following relation:

$$FS = \{[X] \rightarrow [k, Y] : X \in UDS(R) \wedge (k, Y) \in Range((R')^* \setminus \{[0, X]\} \wedge \neg(\exists k' > k\ s.t.\ (k', Y) \in Range(R')^+ \setminus \{[0, X]\})\}, \quad (6)$$

where $UDS(R)$ is a set of ultimate dependence sources calculated as Domain(R)-Range(R); $(R')^{*}\backslash\{[0, X]\}$ means that the domain of relation R'^{*} is restricted to the set including ultimate dependences sources only (elements of this set belong to the first time partition); the constraint $\neg(\exists\ k' > k$ s.t. $(k', Y) \in$ Range$(R')^{+}\backslash\{[0, X]\})$ guarantees that partition k includes only those statement instances whose operands are available, i.e., each statement instance will belong to one time partition only.

It is worth to note that the first element of the tuple, representing the set Range(FS), points out the time of a partition while the last element of that exposes the identifier of the statement whose instance(iteration) is defined by the tuple elements 2 to $n - 1$, where n is the number of the tuple elements of a preprocessed relation. Taking the above consideration into account and provided that the constraints of relation FS are affine, the set Range(FS) is used to generate parallel code applying any well-known technique to scan its elements in the lexicographic order, for example, the techniques presented in papers [2, 11].

The outermost sequential loop of such code scans values of variable k (representing the time of partitions) while inner parallel loops scan independent instances of partition k.

Finally, we expose independent statement instances, that is, those that do not belong to any dependence and generate code enumerating them. According to the free schedule, they are to be executed at time $k = 0$.

6 Implementation

Figure 1 shows the details of the TRACO implementation. Currently, it supports C/C++ programs on its input. A preprocessor, written in the Python language, recognizes loops in a source program and converts them to the format acceptable by the Omega dependence analyzer, Petit, that returns a set of dependence relations representing all the dependences in a loop. Then TRACO recognizes variables to be privatized and/or used for parallel reduction. If privatization and/or reduction remove all dependence relations, parallelization is trivial, all loops can be made parallel. For such a case, TRACO makes the outermost loop to be parallel while the remaining loops to be serial to produce coarse-grained code.

When a set of dependence relations after applying privatization and/or parallel reduction is not empty, the number of synchronization-free slices is calculated. If this number is not equal to one, then data necessary for generating pseudocode representing slices are calculated and forwarded to a pseudocode generator. Otherwise, data necessary for extracting (free) scheduling are prepared and directed to the pseudocode generator. A postprocessor generates parallel code in OpenMP/OpenACC C/C++. Below, we present some details concerned code generation.

Input for the pseudocode generator is a set representing slices or scheduling. For the first case, the first element of the set states for slice representatives, all the following elements, but the last one, describe statement instances of a parametrized slice, and the last one represents a statement identifier, which may be skipped when

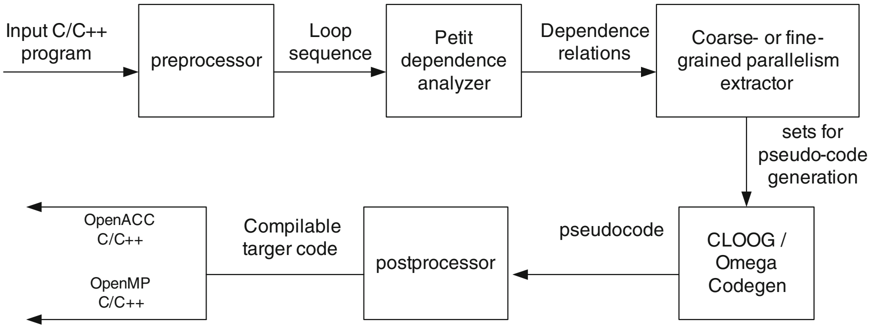


Fig. 1 TRACO organization

(a)

```
S := {[i,j] : 1<=i<=n &&
1 <= j <= n && 2 <= n };
```

(b)

```
if (n >= 2)
  for (int c0 = 1; c0 <= n; c0 += 1)
    for (int c1 = 1; c1 <= n; c1 += 1)
      s1(c0, c1);
```

(c)

```
if (n >= 2)
  #pragma omp parallel for
  for (int c0 = 1; c0 <= n; c0 += 1)
    for (int c1 = 1; c1 <= n; c1 += 1)
      a[c0][c1] = a[c0][c1-1];
```

(d)

```
S := {[k,i,j] : 2+k = i+j &&  1, 3-i <= j <= n
 && 1 <= i <= n && 2 <= n }
```

(e)

```
for (int c0 = 1; c0 < 2 * n - 1; c0 += 1)
 for(int c1=max(-n+c0+ 2,1); c1<=min(c0+1,n); c1+=1)
   s1(c0, c1, c0 - c1 + 2);
```

(f)

```
for (int c0 = 1; c0 < 2 * n - 1; c0 += 1)
 #pragma omp parallel for private(c1)
 for(int c1=max(-n+c0+ 2,1); c1<=min(c0+1,n); c1+=1)
   a[c1][c0-c1+2]=a[c1][c0-c1+2+1]+a[c1+1][c0-c1+2];
```

Fig. 2 Code generation details: **a**–**c** synchronization-free slices; **d**–**f** free scheduling

all dependent statement instances are originated from the same statement. An example set is illustrated in Fig. 2a. In this set, the first element is responsible for slice representatives while the second one together with the first one presents statement instances of a slice. There is no element describing a statement identifier.

Taking such a set as input, CLOOG generates pseudocode Fig. 2b, where by default the outermost loop is to scan slice representatives (this loop is parallel), while the inner loop (serial) enumerates statement instances of the slice with a representative presented by the outermost loop.

Any other code generator, permitting for scanning set elements in the lexicographic order, can be applied in TRACO, for example, the codegen function of Omega or Omega+ [7].

When a set *S* represents scheduling, then the first element of the set is responsible for the time partition representation, all the following elements, but the last one, describe statement instances of a parameterized time partition, and the last one represents a statement identifier, which may be skipped when all statement instances are originated from the same statement. An example set is given in Fig. 2d, where the first element represents time partitions, while the second and third ones are to enumerate statement instances of a particular time partition defined by the first element.

Taking such a set as input, CLOOG generates pseudocode Fig. 2e, where by default the outermost loop scans times (this loop is serial), while the remaining loops (parallel) enumerate statement instances of the time partition for a time represented by the outermost loop.

Compilable OpenMP/OpenACC C/C++ code is produced by means of the post-processor written in Python. It inserts source loop statements with proper index expressions into pseudocode. Original index variables are replaced with variables represented with the tuple elements of a set representing polyhedra taking into account the role of particular tuple elements. For example, provided that the set S on Fig. 2a is associated with the statement a[i][j] = a[i][j − 1], in the pseudo statement s1(c0, c1) on Fig. 2b, variables c0, c1 correspond to variables i, j which are substituted for c0, c1 in the source statement Fig. 2c.

Given the set S in Fig. 2d is associated with the source statement a[i][j] = a[i][j+1]+a[i+1][j], the code generator recognizes that in the pseudocode on Fig. 2e c0 states for time of partitions, c1 corresponds to variable i, while c0 − c1 + 2 corresponds to variable j. So it generates the following statement in the output loop (see Fig. 2f a[c1][c0−c1+2] = a[c0][c0−c1+2+1]+a[c0+1][c0−c1+2]).

Depending on whether pseudocode represents slices or scheduling, the post-processor inserts proper OpenMP pragmas such as *Parallel, For, Critical* and proper clauses to define private and/or reduction variable or OpenACC pragmas such as *Kernel, Data, Loop*.

The source repository of the TRACO compiler is available on the website http://traco.sourceforge.net.

7 Related Work

Different source-to-source compilers have been developed to extract coarse-grained parallelism available in loops. To choose compilers to be compared with TRACO, we have applied the following criteria: it has to (i) be a source-to-source compiler; (ii) support the C language; (iii) produce compilable code in OpenMP/ACC C/C++. The following compilers were chosen to be compared with TRACO: ICC, Pluto, Cetus, and Par4All.

ICC [10]. The Intel Compilers enable threading through automatic parallelization and OpenMP support. With automatic parallelization, the compilers detect loops that can be safely and efficiently executed in parallel and generate multithreaded code.

Pluto [5]. An automatic parallelization tool is based on the polyhedral model [6]. Pluto transforms C programs from source to source for coarse- or fine-grained parallelism and data locality simultaneously. The core transformation framework mainly works to find affine transformations for efficient tiling and fusion, but not limited to those [6]. Pluto does not support variable privatization and reduction recognition.

Par4All [14]. A tool is composed of different components: the PIPS tool [1], the Polylib library [17], and internal parsers. Program transformations available by the compiler include loop distribution, scalar and array privatization, atomizers (reduction of statements to a three-address form), loop unrolling (partial and full), stripmining, loop interchanging, and others.

Cetus [9]. It provides an infrastructure for research on multicore compiler optimizations that emphasizes automatic parallelization by means of the Java API. The compiler targets C programs and supports source-to-source transformations. The tool is limited only to basic transformations: induction variable substitution, reduction recognition, array privatization, pointers, alias, and range analysis.

The compilers, mentioned above, do not based on the transitive closure of dependence graphs.

8 Experimental Study

The goals of experiments were to evaluate such features of TRACO as: effectiveness, the kind of parallelism extracted (coarse- or fine-grained), and efficiency of parallel loops produced. Another goal was to compare these features of TRACO with those demonstrated by the compilers classified for comparison (see Sect. 7). To evaluate the effectiveness of TRACO, we have experimented with NAS Parallel Benchmarks 3.3 (NPB) [15] and Polyhedral Benchmarks 3.2 (PolyBench) [19].

Table 1 presents techniques used by TRACO which acts as follows. First of all, it tries to extract coarse-grained parallelism by applying privatization only, for 39 NAS loops, variable privatization eliminates all dependences, hence loop parallelization is trivial. Next to the remanding benchmarks, the technique presented in Sect. 4 is applied, this results in parallelization of 70 NAS loops. Finally, for the remaining benchmarks, techniques extracting (free) scheduling are applied that yield 22 NAS loops representing fine-grained parallelism. TRACO fails to extract parallelism for the three loops for which each iteration (except the first one) depends on the previous one: *CG_cg*_6, *CG_cg*_8, and *MG_mg*_4.

For the Polybench suite, there exist 48 loops exposing dependences. TRACO is able to parallelize 45 (94 %) loops. One of the LU decomposition loops (*ludcmp_3*) is serial (each iteration depends on the previous one). For the Seidel-2D and Floyd–Warshall loops, TRACO fails to extract any parallelism because all known to us tools permitting for calculating the transitive closure of a dependence representing all the dependences in a loop [12, 20] are not able to produce transitive closure for these loops. There exists a strong need in improving existing algorithms for calculating transitive closure to enhance their effectiveness. 30 PolyBench loops

Table 1 Techniques of loop parallelization

Technique	No. of NAS loops	No. of Polybench loops
Privatization only	39	0
Slicing with privatization and reduction	70	30
Free scheduling	22	15
Loop parallelized	131	45
All loops	134	48

were parallelized by applying algorithms of synchronization-free slices extraction [3]. For 15 PolyBench loops, fine-grained parallelism was found only (the outermost loop is serial).

To check the performance of coarse-grained parallel code, produced with TRACO, we have selected the following four computative heavy NAS loops: *BT_rhs*_1 (Block Tridiagonal Benchmark), *FT_auxfnct.f2p*_2 (Fast Fourier Transform Benchmark), *LU_HP_rhs*_1 (Lower–Upper symmetric Gauss–Seidel Benchmark), linebreak *UA_diffuse*_5 (Unstructured Adaptive Benchmark) and the three PolyBench loops: *fdtd-2d-apml* (FDTD using Anisotropic Perfectly Matched Layer), *symm* (Symmetric Matrix multiply), and *syr2k* (Symmetric Rank-2k Operations).

For each loop qualified for experiments, we have measured execution time, then speedup is calculated. Speedup is a ratio of sequential time and parallel time, $S = T(1)/T(P)$, where P is the number of processors. Experiments were carried out on an Intel Xeon Processor E5645, 12 Threads, 2.4 GHz, 12 MB Cache, and 16 GB RAM.

Figure 3 illustrates code execution times(in seconds) in a graphical way.

To check the performance of fine-grained parallel code, we have selected the two NBP loops: *CG_cg*_4 (Conjugate Gradient Benchmark), *LU_pintgr*_4 (Lower–Upper symmetric Gauss–Seidel Benchmark) and the three PolyBench loops: *adi* (Alternating Direction Implicit solver), *jacobi-2D* (2-D Jacobi stencil computation), and *reg-detect* (2-D Image processing).

Figure 4 presents parallel code speedups. There exist $\log_2 N$, $\log_2(N2 - N1)$, 6**TSTEPS*, 2**STEPS*, and 4**ITER* synchronization points for the fine-grained versions of the *CG_cg*_4, *LU_pintgr*_4, *adi*, *jacobi-2D*, and *reg-detect* benchmarks, respectively. But despite numerous synchronization points, for studied parallel fine-grained loops, positive speedup is achieved ($S > 1$).

Next, we present the comparison of TRACO features with those of the compilers classified for comparison (see Sect. 7). Table 2 presents the effectivenesses of the studied compilers. TRACO is able to parallelize 131 NAS loops and 45 PolyBench loops. Pluto exposes parallelism for 42 NAS and 39 Polybench loops, it does not support variable privatization and parallel reduction, whereas Cetus and Par4All support these transformations and parallelize more NAS loops. ICC parallelizes 56 NAS loops only. Table 2 shows also what kind of parallelism the compilers extract.

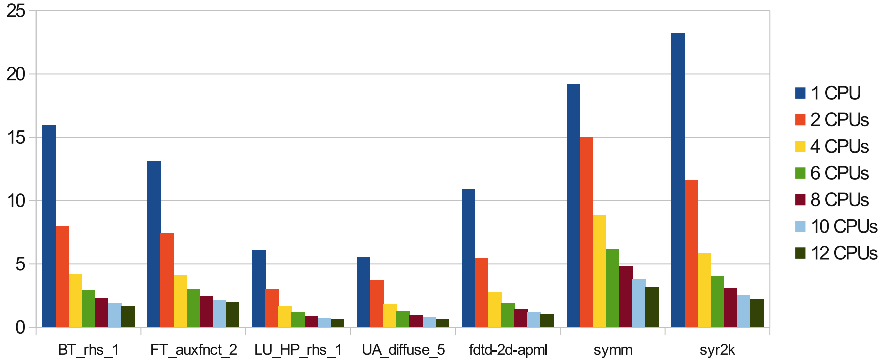


Fig. 3 Times (in seconds) of program loops execution for various numbers of CPUs

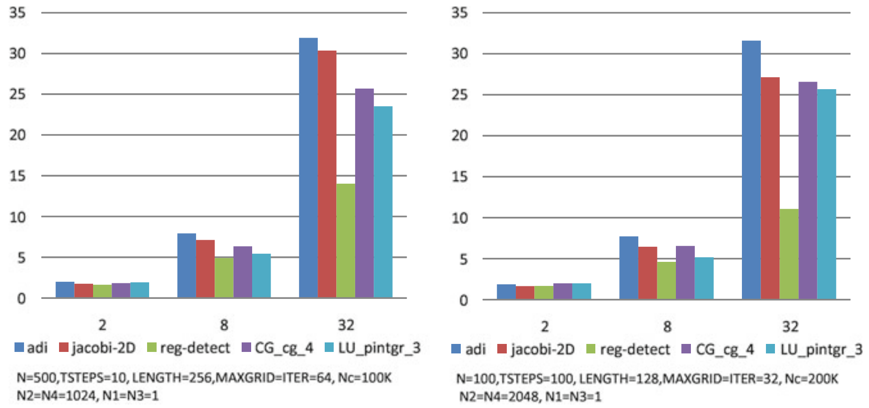


Fig. 4 Speedup of loops representing fine-grained parallelism for various numbers of CUDA cores and loop upper bounds

Table 2 Number of NPB and Polybench loops parallelized by various compilers

Benchmark	Parallelism	TRACO	Pluto	Par4All	Cetus	ICC
NAS	Synchronization-free	109	35	79	107	45
	Fine-grained	22	7	25	19	9
	Total	131	42	104	126	56
Polybench	Synchronization-free	30	29	30	29	28
	Fine-grained	15	10	10	8	9
	Total	45	39	39	38	37

9 Conclusion

We have presented a source-to-source compiler, TRACO, permitting for extracting both coarse- and fine-grained parallelism available in loops represented in the C/C++ language. It implements parallelization algorithms based on the transitive closure of a relation describing all the dependences in a loop and produces compilable parallel OpenMP C/C++ or OpenACC C/C++ code. It is the first compiler that applies the transitive closure of a dependence relation to extract loop parallelism, this enlarges the scope of loop nests which can be parallelized in comparison with existing compilers.

References

1. Amini, M., et al.: PIPS Is not (just) Polyhedral Software. In: International Workshop on Polyhedral Compilation Techniques (IMPACT'11). Chamonix, France April 2011
2. Bastoul, C.: Code generation in the polyhedral model is easier than you think. In: PACT'13 IEEE International Conference on Parallel Architecture and Compilation Techniques. pp. 7–16. Juan-les-Pins September 2004
3. Beletska, A., Bielecki, W., Cohen, A., Palkowski, M., Siedlecki, K.: Coarse-grained loop parallelization: Iteration space slicing vs affine transformations. Parallel Comput. **37**, 479–497 (2011)
4. Bielecki, W., Palkowski, M.: Using free scheduling for programming graphic cards. In: Keller, R., Kramer, D., Weiss, J.P. (eds.) Facing the Multicore - Challenge II. Lecture Notes in Computer Science, LNCS 7174, pp. 72–83. Springer, Berlin (2012)
5. Bondhugula, U., Hartono, A., Ramanujam, J., Sadayappan, P.: A practical automatic polyhedral parallelizer and locality optimizer. In: Proceedings of SIGPLAN Not. 43(6), 101–113 (June 2008), http://pluto-compiler.sourceforge.net
6. Bondhugula, U., et al.: Automatic transformations for communication-minimized parallelization and locality optimization in the polyhedral model. In: Hendren, L. (ed.) Compiler Constructure. Lecture Notes in Computer Science, LNCS 4959, pp. 132–146. Springer, Heidelberg (2008)
7. Chen, C.: Omega+ library. School of Computing University of Utah, (February 2011), http://www.cs.utah.edu/chunchen/omega/
8. Darte, A., Robert, Y., Vivien, F.: Scheduling and Automatic Parallelization. Birkhauser, Boston (2000)
9. Dave, C., Bae, H., Min, S.J., Lee, S., Eigenmann, R., Midkiff, S.: Cetus: A source-to-source compiler infrastructure for multicores. Computer **42**, 36–42 (2009)
10. Intel ® Compilers (2013), http://software.intel.com/en-us/intel-compilers
11. Kelly, W., Maslov, V., Pugh, W., Rosser, E., Shpeisman, T., Wonnacott, D.: The omega library interface guide. Technical report, College Park (1995)
12. Kelly, W., Pugh, W., Rosser, E., Shpeisman, T.: Transitive closure of infinite graphs and its applications. Int. J. Parallel Program. **24**(6), 579–598 (1996)
13. Kennedy, K., Allen, J.R.: Optimizing compilers for modern architectures: a dependence-based approach. Morgan Kaufmann Publishers Inc., CA (2002)
14. Mehdi, A.: Par4All User Guide (2012), http://www.par4all.org
15. NAS benchmarks suite. http://www.nas.nasa.gov
16. Padua, D.A. (ed.): Encyclopedia of Parallel Computing. Springer (2011)
17. Polylib - a library of polyhedral functions, http://icps.u-strasbg.fr/polylib
18. Pugh, W., Wonnacott, D.: An exact method for analysis of value-based array data dependences. In: Sixth Annual Workshop on Programming Languages and Compilers for Parallel Computing. Springer, Berlin (1993)

19. The Polyhedral Benchmark suite (2012), http://www.cse.ohio-state.edu/pouchet/software/polybench/
20. Verdoolaege, S., Cohen, A., Beletska, A.: Transitive closures of affine integer tuple relations and their overapproximations. In: Proceedings of the 18th international conference on Static analysis, SAS'11. pp. 216–232. Springer, Berlin (2011)

On the Multiplication of Biquaternions

Aleksandr Cariow and Galina Cariowa

Abstract We present an efficient algorithm to multiply two arbitrary biquaternions. The schoolbook multiplication of two biquaternions requires 64 real multiplications and 56 real additions. More effective solutions still do not exist. We show how to compute a product of the biquaternions with 24 real multiplications and 56 real additions. During synthesis of the discussed algorithm we use the fact that product of two biquaternions may be represented as a matrix–vector product. The matrix multiplicand that participates in the product calculating has unique structural properties that allow performing its advantageous factorization. Namely this factorization leads to significant reducing of the computational complexity of biquaternion multiplication.

Keywords Biquaternion · Multiplication of biquaternions · Fast algorithms

1 Introduction

The development of theory and practice of data processing as well as the necessity of solving more and more complex problems of theoretical and applied computer science requires using advanced mathematical methods and formalisms. At present hypercomplex numbers [1] are seeing increased application in various fields of digital signal and image processing [2], computer graphics and machine vision [3], telecommunications [4] and in public key cryptography [5]. Preliminary studies show that when solving problems of data processing, quaternions and biquaternions or complexfield quaternions are often used [6].

A. Cariow (✉) · G. Cariowa
West Pomeranian Uniwersity of Technology, Żołnierska 52 Szczecin, Szczecin, Poland
e-mail: atariov@wi.zut.edu.pl

G. Cariowa
e-mail: gtariova@wi.zut.edu.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_35

Among other arithmetical operations in the hypercomplex algebras, multiplication is the most time-consuming one. The reason for this is, because the usual multiplication of these numbers requires $N(N-1)$ real additions and N^2 real multiplication. It is easy to see that the increasing of dimension of hypernumber increases the computational complexity of its multiplication. Therefore, reducing the computational complexity of the multiplication of hypercomplex numbers is an important theoretical and practical task. Efficient algorithms for multiplication of quaternions, octonions, and sedenions already exist [7–9]. No such algorithms for multiplication of the biquaternions have been proposed. In this paper, an efficient algorithm for this purpose is suggested.

2 Preliminary Remarks

A biquaternion is defined as follows [6]:

$$\tilde{b} = b_0 + b_1e_1 + b_2e_2 + b_3e_3 + b_4e_4 + b_5e_5 + b_6e_6 + b_7e_7$$

where $\{b_i\}$, $i = 0, 1, \ldots, 7$ are real numbers, and $\{e_j\}$, $j = 1, 2, \ldots, 7$ are imaginary units whose products are defined by the following table [6]:

×	e_1	e_2	e_3	e_4	e_5	e_6	e_7
e_1	-1	e_3	$-e_2$	e_5	$-e_4$	e_7	$-e_6$
e_2	$-e_3$	-1	e_1	e_6	$-e_7$	$-e_4$	e_5
e_3	e_2	$-e_1$	-1	e_7	e_6	$-e_5$	$-e_4$
e_4	e_5	e_6	e_7	-1	$-e_1$	$-e_2$	$-e_3$
e_5	$-e_4$	e_7	$-e_6$	$-e_1$	1	$-e_3$	e_2
e_6	$-e_7$	$-e_4$	e_5	$-e_2$	e_3	1	$-e_1$
e_7	e_6	$-e_5$	$-e_4$	$-e_3$	$-e_2$	e_1	1

Assume we want to compute the product of two biquaternions $\tilde{b}_3 = \tilde{b}_1\tilde{b}_2$, where

$$\tilde{b}_1 = x_0 + x_1e_1 + x_2e_2 + x_3e_3 + x_4e_4 + x_5e_5 + x_6e_6 + x_7e_7,$$

$$\tilde{b}_2 = b_0 + b_1e_1 + b_2e_2 + b_3e_3 + b_4e_4 + b_5e_5 + b_6e_6 + b_7e_7,$$

$$\tilde{b}_3 = y_0 + y_1e_1 + y_2e_2 + y_3e_3 + y_4e_4 + y_5e_5 + y_6e_6 + y_7e_7.$$

Using “pen and paper” method we can write:

$$\begin{aligned}\tilde{b}_3 = {} & x_0b_0 + x_0b_1e_1 + x_0b_2e_2 + x_0b_3e_3 + x_0b_4e_4 + x_0b_5e_5 + x_0b_6e_6 + x_0b_7e_7\\
& + x_1b_0e_1 + x_1b_1e_1e_1 + x_1b_2e_1e_2 + x_1b_3e_1e_3 + x_1b_4e_1e_4 + x_1b_5e_1e_5 + x_1b_6e_1e_6 + x_1b_7e_1e_7\\
& + x_2b_0e_2 + x_2b_1e_2e_1 + x_2b_2e_2e_2 + x_2b_3e_2e_3 + x_2b_4e_2e_4 + x_2b_5e_2e_5 + x_2b_6e_2e_6 + x_2b_7e_2e_7\\
& + x_3b_0e_3 + x_3b_1e_3e_1 + x_3b_2e_3e_2 + x_3b_3e_3e_3 + x_3b_4e_3e_4 + x_3b_5e_3e_5 + x_3b_6e_3e_6 + x_3b_7e_3e_7\\
& + x_4b_0e_4 + x_4b_1e_4e_1 + x_4b_2e_4e_2 + x_4b_3e_4e_3 + x_4b_4e_4e_4 + x_4b_5e_4e_5 + x_4b_6e_4e_6 + x_4b_7e_4e_7\\
& + x_5b_0e_5 + x_5b_1e_5e_1 + x_5b_2e_5e_2 + x_5b_3e_5e_3 + x_5b_4e_5e_4 + x_5b_5e_5e_5 + x_5b_6e_5e_6 + x_5b_7e_5e_7\\
& + x_6b_0e_6 + x_6b_1e_5e_1 + x_6b_2e_6e_2 + x_6b_3e_6e_3 + x_6b_4e_6e_4 + x_6b_5e_6e_5 + x_6b_6e_6e_6 + x_6b_7e_6e_7\\
& + x_7b_0e_7 + x_7b_1e_7e_1 + x_7b_2e_7e_2 + x_7b_3e_7e_3 + x_7b_4e_7e_4 + x_7b_5e_7e_5 + x_7b_6e_7e_6 + x_7b_7e_7e_7.\end{aligned}$$

Then we have:

$$\begin{aligned}
y_0 &= x_0b_0 - x_1b_1 - x_2b_2 - x_3b_3 - x_4b_4 + x_5b_5 + x_6p_6 + x_7b_7,\\
y_1 &= x_0b_1 + x_1b_0 + x_2b_3 - x_3b_2 - x_4b_5 - x_5b_4 - x_6b_7 + x_7b_6,\\
y_2 &= x_0b_2 - x_1b_3 + x_2b_0 + x_3b_1 - x_4b_6 + x_5b_7 - x_6b_4 - x_7b_5,\\
y_3 &= x_0b_3 + x_1b_2 - x_2b_1 + x_3b_0 - x_4b_7 - x_5b_6 + x_6b_5 - x_7b_4,\\
y_4 &= x_0b_4 - x_1b_5 - x_2b_6 - x_3b_7 + x_4b_0 - x_5b_1 - x_6b_2 - x_7b_3,\\
y_5 &= x_0b_5 + x_1b_4 + x_2b_7 - x_3b_6 + x_4b_1 + x_5b_0 + x_6b_3 - x_7b_2,\\
y_6 &= x_0b_6 - x_1b_7 + x_2b_4 + x_3b_5 + x_4b_2 - x_5b_3 + x_6b_0 + x_7b_1,\\
y_7 &= x_0b_7 + x_1b_6 - x_2b_5 + x_3b_4 + x_4b_3 + x_5b_2 - x_6b_1 + x_7b_0.
\end{aligned}$$

We can see that the schoolbook method of multiplication of two biquaternions requires 64 real multiplications and 56 real additions. In matrix notation, the above relations can be written more compactly as

$$\mathbf{Y}_{8\times 1} = \mathbf{B}_8\mathbf{X}_{8\times 1}, \tag{1}$$

where

$$\mathbf{X}_{8\times 1} = [x_0, x_1, x_2, x_3.x_4, x_5, x_6, x_7]^{\mathrm{T}}, \quad \mathbf{Y}_{8\times 1} = [y_0, y_1, y_2, y_3.y_4, y_5, y_6, y_7]^{\mathrm{T}},$$

$$\mathbf{B}_8 = \left[\begin{array}{cccc|cccc}
b_0 & -b_1 & -b_2 & -b_3 & -b_4 & b_5 & b_6 & b_7\\
b_1 & b_0 & b_3 & -b_2 & -b_5 & -b_4 & -b_7 & b_6\\
b_2 & -b_3 & b_0 & b_1 & -b_6 & b_7 & -b_4 & -b_5\\
b_3 & b_2 & -b_1 & b_0 & -b_7 & -b_6 & b_5 & -b_4\\
\hline
b_4 & -b_5 & -b_6 & -b_7 & b_0 & -b_1 & -b_2 & -b_3\\
b_5 & b_4 & b_7 & -b_6 & b_1 & b_0 & b_3 & -b_2\\
b_6 & -b_7 & b_4 & b_5 & b_2 & -b_3 & b_0 & b_1\\
b_7 & b_6 & -b_5 & b_4 & b_3 & b_2 & -b_1 & b_0
\end{array}\right],$$

The direct realization of (1) requires 64 real multiplications and 56 real additions. We shall present the algorithm, which reduces arithmetical complexity to 24 real multiplications and 56 real additions.

3 The Algorithm

First, we rearrange the columns of the matrix $\mathbf{B}_8$ according to the following rule of ordering (1, 2, 3, 4, 5, 6, 7, 8) → (2, 1, 4, 3, 5, 6, 7, 8). Next, we rearrange the rows of obtained matrix according to the following rule of ordering (1, 2, 3, 4, 5, 6, 7, 8) → (1, 2, 3, 4, 6, 5, 8, 7). The next step of modification of the obtained matrix is to perform some artificial transformations which, as we see latter, will minimize the computational complexity of the final algorithm. Multiply by (-1) the second, 3-s, fifth, and sixth rows of this matrix and then multiply by (-1) the first, fourth, fifth, and sixth columns of the obtained matrix. We can easily see that this transformation leads in the future to minimize the computational complexity of the final algorithm. As a result, we obtain the following matrix:

$$\mathbf{B}'_8 = \left[\begin{array}{cccc|cccc} b_1 & b_0 & -b_3 & b_2 & b_4 & -b_5 & b_6 & b_7 \\ b_0 & -b_1 & b_2 & b_3 & -b_5 & -b_4 & b_7 & -b_6 \\ -b_3 & -b_2 & -b_1 & b_0 & -b_6 & b_7 & b_4 & b_5 \\ -b_2 & b_3 & b_0 & b_1 & b_7 & b_6 & b_5 & -b_4 \\ \hline b_4 & -b_5 & b_6 & b_7 & b_1 & b_0 & -b_3 & b_2 \\ -b_5 & -b_4 & b_7 & -b_6 & b_0 & -b_1 & b_2 & b_3 \\ -b_6 & b_7 & b_4 & b_5 & -b_3 & -b_2 & -b_1 & b_0 \\ b_7 & b_6 & b_5 & -b_4 & -b_2 & b_3 & b_0 & b_1 \end{array}\right].$$

Then we can write

$$\mathbf{B}'_8 = \mathbf{R}_8^{(1)}\mathbf{P}_8^{(1)}\mathbf{B}_8\mathbf{P}_8^{(2)}\mathbf{R}_8^{(2)}$$

and

$$\mathbf{Y}_{8\times1} = \mathbf{B}'_8\mathbf{X}_{8\times1} \tag{2}$$

where

$$\mathbf{P}_8^{(1)} = \left[\begin{array}{cccc|cccc} 1 &&&&&&& \\ & 1 &&&&&& \\ && 1 &&&&& \\ &&& 1 &&&& \\ \hline &&&&& 1 && \\ &&&& 1 &&& \\ &&&&&&& 1 \\ &&&&&& 1 & \end{array}\right], \quad \mathbf{P}_8^{(2)} = \left[\begin{array}{cccc|cccc} & 1 &&&&&& \\ 1 &&&&&&& \\ &&& 1 &&&& \\ && 1 &&&&& \\ \hline &&&& 1 &&& \\ &&&&& 1 && \\ &&&&&& 1 & \\ &&&&&&& 1 \end{array}\right],$$

$$\mathbf{R}_8^{(1)} = \left[\begin{array}{cccc|cccc} 1 &&&&&&& \\ & -1 &&&&&& \\ && -1 &&&&& \\ &&& 1 &&&& \\ \hline &&&& -1 &&& \\ &&&&& -1 && \\ &&&&&& 1 & \\ &&&&&&& 1 \end{array}\right], \quad \mathbf{R}_8^{(2)} = \left[\begin{array}{cccc|cccc} -1 &&&&&&& \\ & 1 &&&&&& \\ && 1 &&&&& \\ &&& -1 &&&& \\ \hline &&&& -1 &&& \\ &&&&& -1 && \\ &&&&&& 1 & \\ &&&&&&& 1 \end{array}\right].$$

Now the matrix $\mathbf{B}'_8$ has a unique block structure[1]:

$$\mathbf{B}'_8 = \left[\begin{array}{c|c} \mathbf{A}_4 & \mathbf{B}_4 \\ \hline \mathbf{B}_4 & \mathbf{A}_4 \end{array}\right],$$

where

$$\mathbf{A}_4 = \left[\begin{array}{cc|cc} b_1 & b_0 & -b_3 & b_2 \\ b_0 & -b_1 & b_2 & b_3 \\ \hline -b_3 & -b_2 & -b_1 & b_0 \\ -b_2 & b_3 & b_0 & b_1 \end{array}\right], \quad \mathbf{B}_4 = \left[\begin{array}{cc|cc} b_4 & -b_5 & b_6 & b_7 \\ -b_5 & -b_4 & b_7 & -b_6 \\ \hline -b_6 & b_7 & b_4 & b_5 \\ b_7 & b_6 & b_5 & -b_4 \end{array}\right].$$

It is easily verified [10] that the matrix with this structure can be factorized, then the computational procedure for multiplication of the biquaternions can be represented as follows:

$$\mathbf{Y}_{8\times 1} = \mathbf{R}_8^{(1)}\mathbf{P}_8^{(1)}\mathbf{W}_8\mathbf{D}_8^{(1)}\mathbf{W}_8\mathbf{P}_8^{(2)}\mathbf{R}_8^{(2)}\mathbf{X}_8^{(2)} \tag{3}$$

where

$$\mathbf{D}_8^{(1)} = \frac{1}{2}(\mathbf{A}_4 + \mathbf{B}_4) \oplus \frac{1}{2}(\mathbf{A}_4 - \mathbf{B}_4), \quad \mathbf{W}_8 = \mathbf{H}_2 \otimes \mathbf{I}_4,$$

$\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$—is the order 2 Hadamard matrix, $\mathbf{I}_N$ is the order N identity matrix, "$\otimes$" and "$\oplus$"—denote the Kronecker product and direct sum of two matrices respectively [10].

$$\mathbf{A}_4 + \mathbf{B}_4 = \left[\begin{array}{cc|cc} b_1 + b_4 & b_0 - b_5 & -b_3 + b_6 & b_2 + b_7 \\ b_0 - b_5 & -b_1 - b_4 & b_2 + b_7 & b_3 - b_6 \\ \hline -b_3 - b_6 & -b_2 + b_7 & -b_1 + b_4 & b_0 + b_5 \\ -b_2 + b_7 & b_3 + b_6 & b_0 + b_5 & b_1 - b_4 \end{array}\right],$$

[1] Using the notation $(\cdot)'$ here and subsequently we shall denote a modified version of the matrix inscribed inside the parentheses.

$$\mathbf{A}_4 - \mathbf{B}_4 = \left[\begin{array}{cc|cc} b_1 - b_4 & b_0 + b_5 & -b_3 - b_6 & b_2 - b_7 \\ b_0 + b_5 & -b_1 + b_4 & b_2 - b_7 & b_3 + b_6 \\ \hline -b_3 + b_6 & -b_2 - b_7 & -b_1 - b_4 & b_0 - b_5 \\ -b_2 - b_7 & b_3 - b_6 & b_0 - b_5 & b_1 + b_4 \end{array}\right],$$

Now we rearrange the columns of the matrix $\mathbf{A}_4 + \mathbf{B}_4$ and $\mathbf{A}_4 - \mathbf{B}_4$ according to the following rule of ordering $(1, 2, 3, 4) \to (4, 2, 3, 1)$. Next, we rearrange the rows of obtained matrices according to the following rule of ordering $(1, 2, 3, 4) \to (2, 3, 1, 4)$. The next step of modification of the obtained matrices is to perform some artificial transformations which, as we see latter, will minimize the computational complexity of the final algorithm. Multiply by (-1) the third rows of both matrices. We can easily see that this transformation leads in the future to minimize the computational complexity of the final algorithm. As a result, we obtain the following matrices:

$$(\mathbf{A}_4 + \mathbf{B}_4)' = \left[\begin{array}{cc|cc} b_3 - b_6 & -b_1 - b_4 & b_2 + b_7 & b_0 - b_5 \\ b_0 + b_5 & -b_2 + b_7 & -b_1 + b_4 & -b_3 - b_6 \\ \hline -b_2 - b_7 & -b_0 + b_5 & b_3 - b_6 & -b_1 - b_4 \\ b_1 - b_4 & b_3 + b_6 & b_0 + b_5 & -b_2 + b_7 \end{array}\right],$$

$$(\mathbf{A}_4 - \mathbf{B}_4)' = \left[\begin{array}{cc|cc} b_3 + b_6 & -b_1 + b_4 & b_2 - b_7 & b_0 + b_5 \\ b_0 - b_5 & -b_2 - b_7 & -b_1 - b_4 & -b_3 + b_6 \\ \hline -b_2 + b_7 & -b_0 - b_5 & b_3 + b_6 & -b_1 + b_4 \\ b_1 + b_4 & b_3 - b_6 & b_0 - b_5 & -b_2 - b_7 \end{array}\right].$$

Indeed, it is easy to see that the matrices $(\mathbf{A}_4 + \mathbf{B}_4)'$, $(\mathbf{A}_4 - \mathbf{B}_4)'$ have the following structures:

$$(\mathbf{A}_4 + \mathbf{B}_4)' = \left[\begin{array}{c|c} \mathbf{A}_2 & \mathbf{B}_2 \\ \hline \mathbf{C}_2 & \mathbf{A}_2 \end{array}\right], \quad (\mathbf{A}_4 - \mathbf{B}_4)' = \left[\begin{array}{c|c} \mathbf{D}_2 & \mathbf{E}_2 \\ \hline \mathbf{F}_2 & \mathbf{D}_2 \end{array}\right],$$

where

$$\mathbf{A}_2 = \left[\begin{array}{c|c} b_3 - b_6 & -b_1 - b_4 \\ \hline b_0 + b_5 & -b_2 + b_7 \end{array}\right], \quad \mathbf{B}_2 = \left[\begin{array}{c|c} b_2 + b_7 & b_0 - b_5 \\ \hline -b_1 + b_4 & -b_3 - b_6 \end{array}\right], \quad \mathbf{C}_2 = \left[\begin{array}{c|c} -b_2 - b_7 & -b_0 + b_5 \\ \hline b_1 - b_4 & b_3 + b_6 \end{array}\right],$$

$$\mathbf{D}_2 = \left[\begin{array}{c|c} b_3 + b_6 & -b_1 + b_4 \\ \hline b_0 - b_5 & -b_2 - b_7 \end{array}\right], \quad \mathbf{E}_2 = \left[\begin{array}{c|c} b_2 - b_7 & b_0 + b_5 \\ \hline -b_1 - b_4 & -b_3 + b_6 \end{array}\right], \quad \mathbf{F}_2 = \left[\begin{array}{c|c} -b_2 + b_7 & -b_0 - b_5 \\ \hline b_1 + b_4 & b_3 - b_6 \end{array}\right].$$

As shown in [10], the matrices having such block structures can also be effectively factorized:

$$\left[\begin{array}{c|c} \mathbf{A}_2 & \mathbf{B}_2 \\ \hline \mathbf{C}_2 & \mathbf{A}_2 \end{array}\right] = (\mathbf{T}_{2\times 3} \otimes \mathbf{I}_2) diag \left[\begin{array}{c} \mathbf{C}_2 - \mathbf{A}_2 \\ \hline \mathbf{B}_2 - \mathbf{A}_2 \\ \hline \mathbf{A}_2 \end{array}\right] (\mathbf{T}_{3\times 2} \otimes \mathbf{I}_2) \tag{4}$$

$$\left[\begin{array}{c|c}\mathbf{D}_2 & \mathbf{E}_2 \\ \hline \mathbf{F}_2 & \mathbf{D}_2\end{array}\right] = (\mathbf{T}_{2\times 3} \otimes \mathbf{I}_2) diag \left[\begin{array}{c}\mathbf{F}_2 - \mathbf{D}_2 \\ \hline \mathbf{E}_2 - \mathbf{D}_2 \\ \hline \mathbf{D}_2\end{array}\right] (\mathbf{T}_{3\times 2} \otimes \mathbf{I}_2) \tag{5}$$

Substituting (4), (5) in (3) we can write:

$$\mathbf{Y}_{8\times 1} = \mathbf{R}_8^{(1)}\mathbf{P}_8^{(1)}\mathbf{W}_8\mathbf{W}_{8\times 12}\mathbf{D}_{12}\mathbf{W}_{12\times 8}\mathbf{W}_8\mathbf{R}_8^{(2)}\mathbf{P}_8^{(2)}\mathbf{X}_{8\times 1} \tag{6}$$

where

$$\mathbf{W}_{8\times 12} = \mathbf{I}_2 \otimes (\mathbf{R}_4^{(1)}\mathbf{P}_4^{(1)}\mathbf{W}_{4\times 6}),$$

$$\mathbf{W}_{4\times 6} = (\mathbf{T}_{2\times 3} \otimes \mathbf{I}_2) = \begin{bmatrix} & & 1 & & 1 & \\ & & & 1 & & 1 \\ 1 & & & & 1 & \\ & 1 & & & & 1 \end{bmatrix}, \quad \mathbf{P}_4^{(1)} = \begin{bmatrix} & 1 & & \\ & & 1 & \\ 1 & & & \\ & & & 1 \end{bmatrix}, \quad \mathbf{T}_{2\times 3} = \begin{bmatrix} & 1 & 1 \\ 1 & & 1 \end{bmatrix},$$

$$\mathbf{R}_4^{(1)} = \begin{bmatrix} 1 & & & 1 \\ & 1 & & \\ & & -1 & \\ & & & 1 \end{bmatrix}, \quad \mathbf{W}_{12\times 8} = \mathbf{I}_2 \otimes (\mathbf{W}_{6\times 4}\mathbf{P}_4^{(2)}), \quad \mathbf{P}_4^{(2)} = \begin{bmatrix} & & & 1 \\ & 1 & & \\ & & 1 & \\ 1 & & & \end{bmatrix},$$

$$\mathbf{W}_{6\times 4} = (\mathbf{T}_{3\times 2} \otimes \mathbf{I}_2) = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \\ 1 & & 1 & \\ & 1 & & 1 \end{bmatrix}, \quad \mathbf{T}_{3\times 2} = \begin{bmatrix} 1 & \\ & 1 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{D}_{12} = \frac{1}{2} diag \left[\begin{array}{c}(\mathbf{C}_2 - \mathbf{A}_2) \\ \hline (\mathbf{B}_2 - \mathbf{A}_2) \\ \hline \mathbf{A}_2 \\ \hline (\mathbf{F}_2 - \mathbf{D}_2) \\ \hline (\mathbf{E}_2 - \mathbf{D}_2) \\ \hline \mathbf{D}_2\end{array}\right],$$

$$\mathbf{C}_2 - \mathbf{A}_2 = \left[\begin{array}{c|c} -b_2 - b_3 + b_6 - b_7 & -b_0 + b_1 + b_4 + b_5 \\ \hline -b_0 + b_1 - b_4 - b_5 & b_2 + b_3 + b_6 - b_7 \end{array}\right],$$

$$\mathbf{B}_2 - \mathbf{A}_2 = \left[\begin{array}{c|c} b_2 - b_3 + b_6 + b_7 & b_0 + b_1 + b_4 - b_5 \\ \hline -b_0 - b_1 + b_4 - b_5 & b_2 - b_3 - b_6 - b_7 \end{array}\right],$$

$$\mathbf{F}_2 - \mathbf{D}_2 = \left[\begin{array}{c|c} -b_2 - b_3 - b_6 + b_7 & -b_0 + b_1 - b_4 - b_5 \\ \hline -b_0 + b_1 + b_4 + b_5 & b_2 + b_3 - b_6 + b_7 \end{array}\right],$$

$$\mathbf{E}_2 - \mathbf{D}_2 = \left[\begin{array}{c|c} b_2 - b_3 - b_6 - b_7 & b_0 + b_1 - b_4 + b_5 \\ \hline -b_0 - b_1 - b_4 + b_5 & b_2 - b_3 + b_6 + b_7 \end{array}\right],$$

Introduce the following notation:

$$\begin{aligned}
&c_0 = 1/2(-b_2 - b_3 + b_6 - b_7), \quad c_1 = 1/2(-b_0 + b_1 + b_4 + b_5), \quad c_2 = 1/2(-b_0 + b_1 - b_4 - b_5),\\
&c_3 = 1/2(b_2 + b_3 + b_6 - b_7), \quad c_4 = 1/2(b_2 - b_3 + b_6 + b_7), \quad c_5 = 1/2(b_0 + b_1 + b_4 - b_5),\\
&c_6 = 1/2(-b_0 - b_1 + b_4 - b_5), \quad c_7 = 1/2(b_2 - b_3 - b_6 - b_7), \quad c_8 = 1/2(b_3 - b_6),\\
&c_9 = 1/2(-b_1 - b_4), \quad c_{10} = 1/2(b_0 + b_5), \quad c_{11} = 1/2(-b_2 + b_7),\\
&c_{12} = 1/2(-b_2 - b_3 - b_6 + b_7), \quad c_{13} = 1/2(-b_0 + b_1 - b_4 - b_5),\\
&c_{14} = 1/2(-b_0 + b_1 + b_4 + b_5), \quad c_{15} = 1/2(b_2 + b_3 - b_6 + b_7),\\
&c_{16} = 1/2(b_2 - b_3 - b_6 - b_7), \quad c_{17} = 1/2(b_0 + b_1 - b_4 + b_5),\\
&c_{18} = 1/2(-b_0 - b_1 - b_4 + b_5), \quad c_{19} = 1/2(b_2 - b_3 + b_6 + b_7),\\
&c_{20} = 1/2(b_3 + b_6), \quad c_{21} = 1/2(-b_1 + b_4), \quad c_{22} = 1/2(b_0 - b_5), \quad c_{23} = 1/2(-b_2 - b_7).
\end{aligned}$$

Using the above notations and combining partial decompositions in a single computational procedure we finally can write the following:

$$\mathbf{Y}_{8\times1} = \mathbf{R}_8^{(1)}\mathbf{P}_8^{(1)}\mathbf{W}_8\mathbf{W}_{8\times12}\mathbf{W}_{12\times24}\mathbf{D}_{24}^{(3)}\mathbf{W}_{24\times12}\mathbf{W}_{12\times8}\mathbf{W}_8\mathbf{R}_8^{(2)}\mathbf{P}_8^{(2)}\mathbf{X}_{8\times1} \tag{7}$$

where

$$\mathbf{D}_{24}^{(3)} = diag(c_0, c_1, \ldots, c_{23}), \quad \mathbf{W}_{12\times24} = \mathbf{I}_6 \otimes \mathbf{K}_{2\times4}, \quad \mathbf{K}_{2\times4} = \left[\begin{array}{cc|cc} 1 & & 1 & \\ \hline & 1 & & 1 \end{array}\right],$$

$$\mathbf{W}_{24\times12} = \mathbf{I}_6 \otimes \tilde{\mathbf{K}}_{4\times2}, \quad \tilde{\mathbf{K}}_{4\times2} = \left[\begin{array}{c|c} 1 & \\ 1 & \\ \hline & 1 \\ & 1 \end{array}\right].$$

We can see that the ordinary approach to calculation of elements $\{c_k\}, k = 0, 1, \ldots, 23$ of the matrix $\mathbf{D}_{24}$ requires 56 additions. It is easy to see that the relations for calculation of $\{c_k\}$ contain repeated algebraic sums. Therefore, the number of additions necessary to calculate these elements can be significantly reduced. So, it is easy to verify that the elements $\{c_k\}$, $k = 0, 1, \ldots, 23$ can be calculated using the following rationalized matrix–vector procedure:

$$\mathbf{C}_{24\times1} = \mathbf{P}_{24\times16}\tilde{\mathbf{W}}_{16}\mathbf{P}_{16\times8}\mathbf{P}_8^{(2)}\tilde{\mathbf{D}}_8\mathbf{W}_8^{(0)}\mathbf{P}_8^{(1)}\mathbf{B}_{8\times1} \tag{8}$$

where

$$\mathbf{P}_{24\times 16} = \begin{bmatrix}
 & & & & -1 & & & & & & & & & & & \\
 & & & 1 & & & & & & & & & & & & \\
 & -1 & & & & & & & & & & & & & & \\
 & & & & & & 1 & & & & & & & & & \\
 & & & & & 1 & & & & & & & & & & \\
 & & 1 & & & & & & & & & & & & & \\
-1 & & & & & & & & & & & & & & & \\
 & & & & & & & -1 & & & & & & & & \\
 & & & & & & & & & & & & & 1 & & \\
 & & & & & & & & & & -1 & & & & & \\
 & & & & & & & & 1 & & & & & & & \\
 & & & & & & & & & & & & & & & -1 \\
 & & & & & & -1 & & & & & & & & & \\
 & -1 & & & & & & & & & & & & & & \\
 & & & 1 & & & & & & & & & & & & \\
 & & & & 1 & & & & & & & & & & & \\
 & & & & & & & -1 & & & & & & & & \\
1 & & & & & & & & & & & & & & & \\
 & & -1 & & & & & & & & & & & & & \\
 & & & & & 1 & & & & & & & & & & \\
 & & & & & & & & & & & & & & 1 & \\
 & & & & & & & & & -1 & & & & & & \\
 & & & & & & & & & & & 1 & & & & \\
 & & & & & & & & & & & & -1 & & &
\end{bmatrix},$$

$$\mathbf{P}_8^{(1)} = \begin{bmatrix}
1 & & & & & & & \\
 & & & & & 1 & & \\
 & 1 & & & & & & \\
 & & & & 1 & & & \\
 & & 1 & & & & & \\
 & & & & & & & 1 \\
 & & & 1 & & & & \\
 & & & & & & 1 &
\end{bmatrix}, \quad \mathbf{P}_8^{(2)} = \begin{bmatrix}
1 & & & & & & & \\
 & & & 1 & & & & \\
 & & 1 & & & & & \\
 & 1 & & & & & & \\
 & & & & 1 & & & \\
 & & & & & & & 1 \\
 & & & & & & 1 & \\
 & & & & & 1 & &
\end{bmatrix},$$

$$\mathbf{B}_{8\times 1} = [b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7]^{\mathrm{T}}, \quad \mathbf{C}_{24\times 1} = [c_0, c_1, \ldots, b_{23}]^{\mathrm{T}}, \quad \tilde{\mathbf{D}}_8 = \frac{1}{2}\mathbf{I}_8,$$

$$\mathbf{W}_8^{(0)} = \mathbf{I}_4 \otimes \mathbf{H}_2, \quad \tilde{\mathbf{W}}_{16} = \mathbf{W}_8^{(0)} \oplus \mathbf{I}_8.$$

Figure 1 shows a data flow diagram, which describes the fast algorithm for computation of the biquaternions product and Fig. 2 shows a data flow diagram of the process for calculating the vector $\mathbf{C}_{24\times 1}$ elements. In this paper, data flow diagrams are oriented from left to right. Straight lines in the figures denote the operations

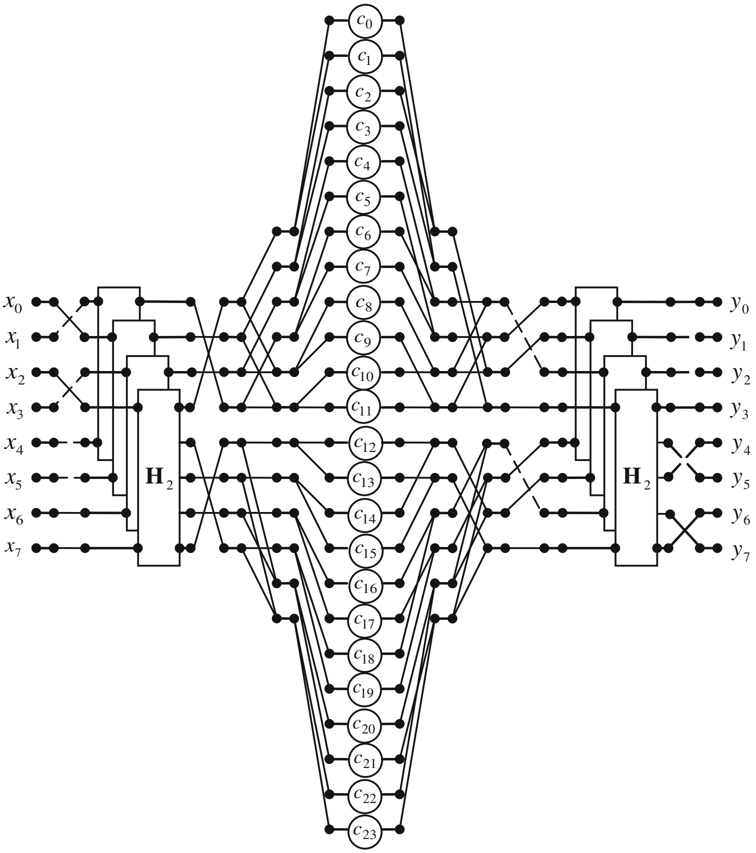


Fig. 1 Data flow diagram of rationalized algorithm for multiplication of two biquaternions

of data transfer. Points where lines converge denote summation. The dash-dotted lines indicate the sign change operation. We use the usual lines without arrows on purpose, so as not to clutter the picture. The circles in these figures show the operation of multiplication by a variable (or constant) inscribed inside a circle. In turn, the rectangles indicate the matrix–vector multiplications with the order 2 Hadamard matrices. As follows from Fig. 2, calculation of elements of diagonal matrix $\tilde{\mathbf{D}}_8$ requires performing only trivial multiplications by the power of 2. Such operations may be implemented as primitive shift operations, which have simple realization and hence may be neglected during computational complexity estimation.

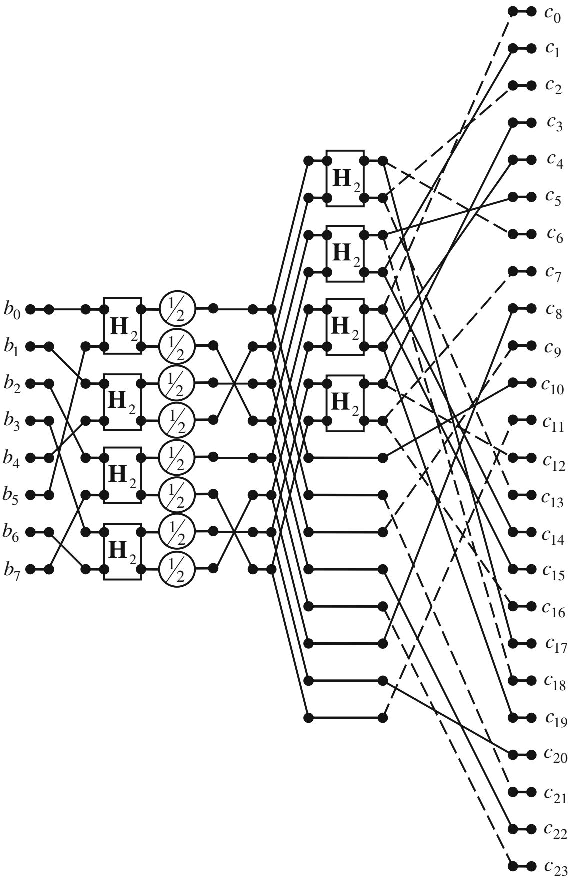


Fig. 2 The data flow diagram describing the process of calculating elements of the vector $\mathbf{C}_{24\times 1}$ in accordance with the procedure (8)

4 Estimation of Computational Complexity

We calculate how many real multiplications (excluding multiplications by power of two) and real additions are required for realization of the proposed algorithm, and compare it with the number of operations required for a direct calculation of matrix–vector product in Eq. (1). As already mentioned, the number of real multiplications required using the proposed algorithm is 24. Thus using the proposed algorithm the number of real multiplications to calculate the biquaternion product is significantly reduced. On the other hand the number of real additions required using our algorithm is 56. Thus, our proposed algorithms saves 40 multiplications while the number of additions is the same as in the naive method. Therefore, the total number of arithmetic operations for the proposed algorithm is approximately 33 % less than that of the direct evaluation. It should be noted that in many practical applications, one of the biquaternions to be multiplied contains constant coefficients. In this case, the diagonal matrix elements can be precomputed once and stored in memory. This would reduce the number of additions in the proposed algorithm to 40.

5 Conclusion

In this paper, we have presented an original algorithm which allows multiplying two arbitrary biquaternions with reduced multiplicative complexity. As a result of streamlining the number of multiplications required to calculate the biquaternion product, it is reduced from 64 to 24. Furthermore, the total number of arithmetic operations decreased by 40 compared with the schoolbook method of calculations. Therefore, the proposed algorithm is better than the direct algorithm, even in terms of its software implementation on a conventional general purpose computer.

References

1. Kantor, I., Solodovnikov, A.: Hypercomplex numbers: an elementary introduction to algebras. Springer, New York (2011). Softcover reprint of the original 1st ed. 1989 edition
2. Alfsmann, D., Göckler, H.G., Sangwine, S.J., Ell, T.A.: Hypercomplex algebras in digital signal processing: benefits and drawbacks (tutorial). In: Proceedings of the EURASIP 15th European Signal Processing Conference, pp. 1322–1326. Poznań (2007)
3. Moxey, C.E., Sangwine, S.J., Ell, T.A.: Hypercomplex correlation techniques for vector images. IEEE Trans. Signal Process. **51**, 1941–1953 (2003)
4. Calderbank, R., Das, S., Al Dhahir, N., Diggavi, S.: Construction and analysis of a new quaternionic space-time code for 4 transmit antennas. Commun. Inf. Syst. **5**, 97–122 (2005)
5. Malekian, E., Zakerolhosseini, A., Mashatan, A.: QTRU: quaternionic version of the NTRU public-key cryptosystems. Int. J. Inf. Secur. **3**, 29–42 (2011)
6. Sangwine, S.J., Ell, T.A., Le Bihan, N.: Fundamental representations and algebraic properties of biquaternions or complexified quaternions. Appl. Clifford Algebras **21**(3), 607–636 (2010)
7. Makarov, O.M.: An algorithm for the multiplication of two quaternions. Zh. Vychisl. Mat. Mat. Fiz. **17**(6), 1574–1575 (1977)
8. Cariow, A., Cariowa, G.: Algorithm for multiplying two octonions. Radioelectronics and Communications Systems, pp. 464–473. Allerton Press, Inc., New York (2012)
9. Cariow, A., Cariowa, G.: An algorithm for fast multiplication of sedenions. Inf. Process. Lett. **113**, 324–331 (2013)
10. Tariov, A.: Algorytmiczne aspekty racjonalizacji obliczeń w cyfrowym przetwarzaniu sygnałów, Wydawnictwo Uczelniane ZUT (2011)

Model of Collaborative Data Exchange for Inland Mobile Navigation

Tomasz Hyla, Natalia Wawrzyniak and Witold Kazimierski

Abstract Current mobile navigation systems increasingly rely on users' input and networking. Inland traffic participants use many different sensors for navigational purposes which can be used to acquire missing information or to verify data provided in Electronic Navigational Charts or by other available information services. MOBINAV system is being developed mainly for recreational users of inland waters. It combines marine achievements in fields of advanced ECDIS systems with inland and leisure specifics needed to ensure a complete picture of the navigational situation. In this paper we present a full model of user data exchange obtained by their mass collaboration. Main assumptions, types of information, model description, and its verification method are presented.

Keywords Inland navigation · Data-exchange · Mass collaboration

1 Introduction

Currently existing navigation support applications designed for inland recreational users try to implement cartographical and routing patterns, which superbly work in on-land mapping, for inland waters navigation purposes. Due to the different specifics of on-land and on-water travel it is not possible to make a direct adaptation. However,

T. Hyla
Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, Szczecin, Poland
e-mail: thyla@zut.edu.pl

N. Wawrzyniak (✉)
Marine Technology Ltd, Szczecin, Poland
e-mail: n.wawrzyniak@marinetechnology.pl

W. Kazimierski
Institute of Geoinformatics, Maritime University of Szczecin, Szczecin, Poland
e-mail: w.kazimierski@am.szczecin.pl

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_36

the achievements in drawing from users' mass collaboration to make systems data constantly up-to date are worth applying in inland navigation.

The main navigation systems used on waters are Electronic Charts Display Systems (ECDIS), which exist also in their adapted form for inland waterways [1, 2]. Their basic function is to present information about an ongoing journey backed with both spatial and navigational information in the form of electronic charts. Inland ECDIS were also designed to integrate data from various on-board sensors (GPS, compass, echosounder) with the data derived from available River Information Services (RIS), such as AIS or NtS [3–5]. These systems are dedicated for commercial use on merchant ships and through its standardized, and thus rigid form, they do not have sufficient flexibility to meet the needs of recreational users [6]. Leisure craft of inland waters use mainly mobile technologies to navigate on desired areas exploring any kind of Web map service available. Due to the latest growth in development of mobile mapping applications, such applications are created also for these specific recipients. However, they are usually limited to vessels positioning against maps background and providing weather information.

MOBINAV system [7] consists of mobile application and Web services dedicated for inland recreational units, which is being developed currently by Marine Technology Ltd. It combines basic features of ECDIS systems with achievements of mobile cartography and user-centered design [8]. It uses collections of spatial data in the form of inland ENCs and navigational data provided by whatever devices and means possible on board (e.g., GPS, compass, echosounder). It allows loading and using any charts available for required area in a standardized format. Any additional, nonspatial information that aid navigation can be acquired from accessible online services, e.g., River Information System [9], or entered manually. Moreover, MOBINAV allows its users to exchange data between each other and to support system and other users by their cooperation [10].

In this paper we propose a model of data exchange used to support systems with information acquired by individual participants of inland waterways. Our approach is based on using mass collaboration to provide system with up to date information.

The paper is organized as follows. Section 2 contains a description of problems related to modeling mass users' data exchange. Section 3 contains our new model definition with its implementation in the MOBINAV system. Section 4 describes model verification by scenarios. The paper ends with conclusions.

2 Background

Users' input in co-creating, developing, and maintaining modern systems is an interesting solution worth consideration in designing any support decision systems. However, allowing users to share and update information in navigational systems, where safety and reliability of data are priorities, needs close preparation of data model before taking such an approach.

2.1 Users Input in Systems Development

Social collaboration is a cooperation of a group of people, who usually share similar interests, have mutual understanding for each other, and are united in achieving a common goal or have a certain mission to fulfill [10, 11]. Together with modern technologies and networking it became a powerful tool especially in developing extensive systems with fast alternating data, provided that their users are willing to contribute. It works spectacularly well for information integration problems. Its use in crowdsourcing of spatial data and mapping systems is priceless—both literally and figuratively—allowing projects as Open Street Map [12] to unwind.

It is also used in many navigation systems in various ways (e.g., indoor positioning, or lakes mapping [13]). But its most interesting application, from inland navigation point of view, is users' mass collaboration applied in car navigation systems to share information about current situation on road [14].

For every existing on-water navigation system the most important issue is safety [15]. The inland mobile navigation environment is relatively quickly alternating, the leisure craft number is growing, and inland charts in many areas are rarely updated. Sometimes not all of the area is charted and the access to information about water levels and navigational situation varies much depending on the location. Detailed and up-to-date bathymetric information is often a rarity [16–18].

The MOBINAV system is based on both approaches to ensure users' safety. The traditional one is to aid its users by providing them with most up-to-date reliable information in the form of a chart and navigational analysis. The other is giving them the tools and means to share their own data and allow reporting to support system and to the other waterways participants or authorities. To achieve this goal a dedicated model of collaborative data exchange had to be designed. User data types and their significances differ greatly and the distinction has to be made to ensure validity preservation of information strictly concerning users immediate safety. Therefore the information had to be prioritized. Some would require storing and later post-processing, others direct notification or participants' confirmations (i.e., in voting process).

The most flexible way to ensure data control and transfer to other systems (if needed) is through XML technology and Simple Object Access Protocol (SOAP) [19]. Its application allows to use push and pop services and to transfer data to other systems by SOAP interfaces. This technology is applied in existing RIS systems deployed across Europe according to the EU RIS Directive (2005/44/EC).

2.2 MOBINAV Data Model

MOBINAV system consists of two logical parts. First, is a mobile-side that consists of mobile devices with an MOBINAV application, which allows to read charts, navigate, and collaborate with other systems users. The second, server-side is a

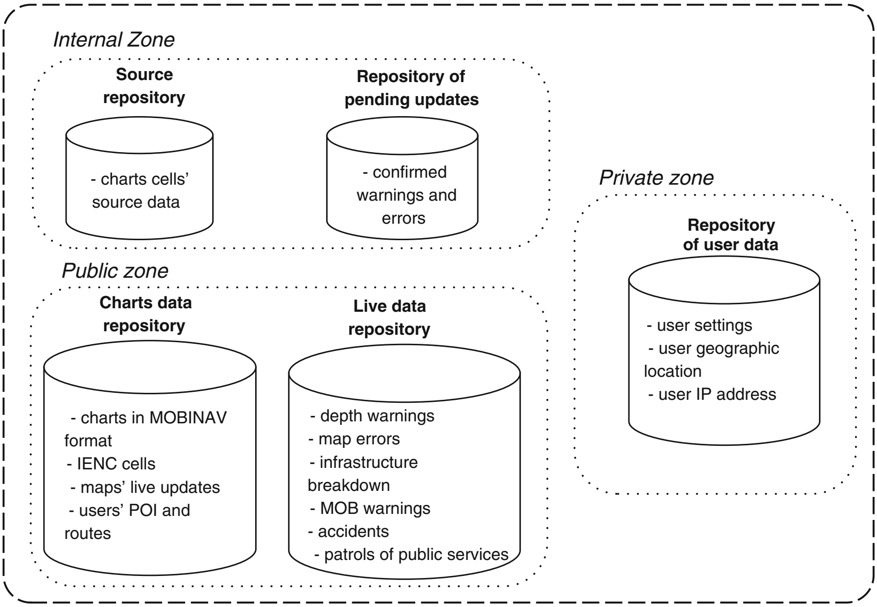


Fig. 1 MOBINAV server-side data model

system backbone. It is placed on servers in services provider data center. The server-side is responsible for providing updated charts (including MOBINAV proprietary format) and enables users to share information they found interesting. Starting from custom Point of Interest (POI) and ending with emergency information like Man Over Board (MOB) alarm.

On the server-side of the system (see Fig. 1) there are three types of data storage zones. The internal zone is used to store data not directly available to users—mainly data used to build charts in MOBINAV proprietary format and verified data from users with permanent changes. The system operator, once every *n* days, publishes from that data a new version of charts. The updated charts are placed in the data charts repository. Charts data repository is in a public zone. The public zone contains information generally available to all users. Besides up-to-date charts, it contains POI and routes shared by users, IENC cells available publicly and so-called live updates. Live update is a set of data reported by users (and partially verified by them) that enables to facilitate navigation and increase safety. Live data gathered from users is stored in a separate repository and after some processing is transferred to the chart data repository. Live data includes navigation warnings such as depth warnings, notifications of accidents, or patrols of public services (like police or boarder guard).

The private zone contains only user data repository. It contains information about users (IP address, GPS location) that can be used by the system only in some emergency situations (e.g., MOB). Other information is user settings that allow users to store their private configuration in the cloud to facilitate usage of many mobile devices by a single user. Except in emergency situations, data from public zone are provided to the users using Web services and service-oriented architecture.

3 Data Exchange Model of User-Acquired Data

In this section we present a data exchange model that can be used in information systems that use a collaboration of users to collectively gather data. We illustrate the data model with the case of the MOBINAV system.

3.1 Model Definition

Definition 1 The Collaborative Data Exchange (CDE) model consists of Assumptions, Categories, and Flows $M = (A, C, F)$, where $A = (A1, A2, A3, A4, A5, A6)$, $C = (C1, C2, C3)$ and $F = (F1, F2, F3)$.

The model assumes the following:

A1. Data are gathered only by users.
A2. Data are stored in a central information system.
A3. System provides data on request except in emergency situations.
A4. Central information system can send gathered data to other information systems only if data are anonymized.
A5. Data from users have an unconfirmed status until confirmed by other users.
A6. Data are transmitted in near-real time using Internet connection.

Each data received from the users is categorized according to acquisition method, safety impact and validity in the following way:

C1. acquisition method = ($c1\_1$ manual, $c1\_2$ automatic)
C2. safety impact = ($c2\_1$ high, $c2\_2$, medium, $c2\_3$ low)
C3. validity = ($c3\_1$ temporary, $c3\_2$ permanent)

The following data flows can be identified:

F1. user to system
F2. system to user
F3. system to other system

3.2 MOBINAV CDE Model Implementation

The MOBINAV collective data (live data) can be divided into categories as shown in Table 1.

Table 1 MOBINAV live data

Live Data	Acquisition method		Safety impact			Validity	
	Manual	Automatic	High	Medium	Low	Temporary	Permanent
Accidents	×			×		×	
Chart errors	×				×		×
Depth warnings		×		×			×
Infrastructure breakdown	×				×	×	
MOB warnings	×		×			×	
Patrols of public services	×			×		×	

The model was implemented in a MOBINAV system. Figure 2 presents a high-level data flow diagram, which visualize data flows inside the MOBINAV. The data flow diagram presents only a view of the system from the perspective of the live data (collaborative data).

The mobile-side contains three processes related to the live data: enable MOB function, trigger manually, and monitor depth. Monitor depth is the only process that sends data automatically to the system. The process is responsible for constant depth control (it does not mean that the process constantly sends data; it sends data only if several condition are met. It is based on decision support algorithms [6]). Other live data messages are sent on events generated by a human operator and they are sent to the system using the same path (data flow). The only exception is man over board function, which goes to the system on separate data flow. Because of its safety impact, the system takes another set of actions including immediate push notification to other users in the area.

The main process in the server-side of the system is the receive request process. It is responsible for dividing live data depending on its category into: automatically acquired (update depth warnings), manually acquired temporary information (update temporary information) or into manually acquired permanent information (update permanent information). The process converts data into a required form and saves it in the live data repository. MOB requests are received by the emergency warnings process. The process gets the IP address of users in the nearby area and forwards warning to them. Also, MOB warning is sent to the live data repository.

Two processes monitor changes in the live data repository: detect updates and send to RIS. The first one is responsible for detecting permanent changes and stores them in the pending updates repository, while the second one is responsible for detecting and sending changes useful from the perspective of the RIS system. The mobile MOBINAV application can retrieve the live data for a given region by sending its geographic position to the prepare live data for a given region process.

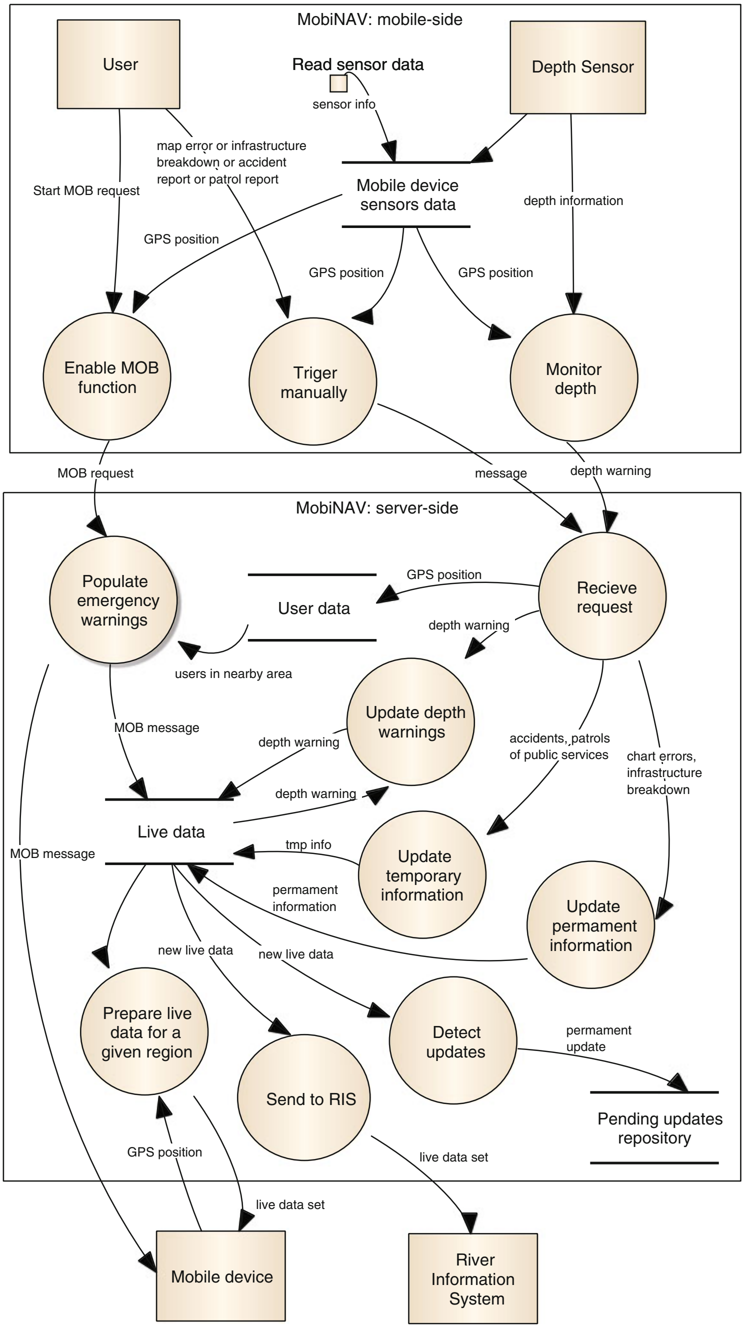


Fig. 2 MOBINAV data flow diagram from "live data" view

4 Scenario Analysis

Series of scenarios were created to verify MOBINAV CDE model implementation with CDE model definition. Below, we present one exemplary use scenario with analysis of its relation to the model from Definition 1 and advantages of using the CDE model.

Scenario. User sails a yacht on a river sometime after the area underwent a flood. The flood causes charts to be imprecise. The user uses system with a calibrated echosounder (MOBINAV provides a Wi-Fi module to which an echosounder is connected, which transmits depth readings to a mobile device). MOBINAV mobile application compares actual depth readings with those from the chart and if difference is less than given threshold, the readings are sent to the server-side of the MOBINAV system using XML message as shown in Fig. 3. The system processes the data and updates the live updates repository. User receives live updates from the server. The warnings are displayed on their charts.

CDE model conformance. The scenario does not violate any assumptions from a set of assumptions A. Depth warnings belong to the category: $c1\_2$ (acquisition method—automatic) $c2\_2$ (safety impact—medium) and $c3\_2$ (validity—permanent). The scenario use data flows $F1$ (depth warning: from *Monitor depth* to *Receive request*), $F2$ (a live data set: *Prepare live data for a given region* to *Mobile device*), $F3$ (a live data set: from *Send to RIS* to *River Information System*).

Advantages of using CDE model. The user can see depth warnings from other users which increases his safety and navigational capabilities. This would not be

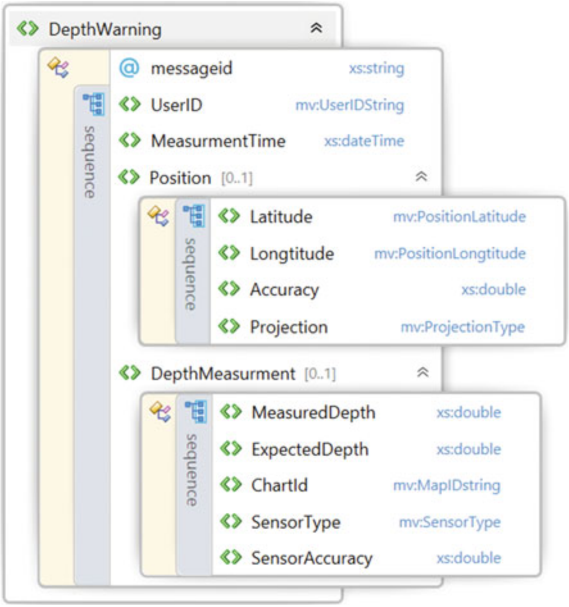


Fig. 3 XML schema of a depth warning message

possible without user collaboration, due to high cost of constant measurement by specialized hydrographic unit. Also, a RIS operator can see warnings from users and if they repeat, he can send specialized ship to measure the uncertain area. This accelerates initial recognition of possible depth problems.

5 Conclusion

Inland navigation systems are based on spatial and navigational information [20], which amounts to rapid growth disproportional to their expected precision and validity. Crowdsourcing and networking are the best ways to provide system with extensive information, incredibly costly, or even unobtainable otherwise. Leisure craft community is generally an easily socializing, cooperative group of people, sharing similar interest, understanding the need for information and experience exchange for the common good. Because this community became MOBINAV systems' target users, the idea of their collaboration in co-creating systems quality seemed natural.

The main purpose of our research was to define a model of data-exchange which would allow sharing and managing various data provided by users' mass collaboration. Definition of such model allows to simplify the process of similar system development by creating a design pattern for future use. The solution itself allows to reduce costs of developing such systems by using simpler architecture, needing less maintenance and limiting data acquisition costs. Collaboration of systems users can also provide more interesting additional functionality for navigational purposes.

Acknowledgments This scientific research work is supported by the National Centre for Research and Development (NCBiR) of Poland (grant No. LIDER/039/693/L-4/12/NCBR/2013) in 2013–2016.

References

1. Pillich, B., Schack, C.: Next generation ECDIS for commercial and military uses. OCEANS '02 MTS/IEEE **2**, 1025–1032 (2002)
2. Pietrzykowski, Z., Borkowski, P., Wolejsza, P.: Marine integrated navigational decision support system. In: Book Series: Communications in Computer and Information Science, 12th International Conference on Transport Systems Telematics, vol. 329, pp. 284–292 (2012)
3. Kazimierski, W., Wawrzyniak, N.: Modification of ECDIS interface for the purposes of geoinformatic system for port security. Annuals Navig. **20**, 51–70 (2013). Gdansk
4. Kazimierski, W.: Problems of Data Fusion of Tracking Radar and AIS for the Needs of Integrated Navigation Systems at Sea. In: Rohling, H. (ed.) Book Series: International Radar Symposium Proceedings, 14th International Radar Symposium (IRS), vols. 1 and 2, pp. 270–275. Dresden (2013)
5. Pfliegl, R., Back, A.: Increasing the attractiveness of inland waterway transport with e-transport river information services. Transp. Res. Rec. **1963**, 15–22 (2006)

6. Lubczonek, J., Stateczny, A.: Aspects of spatial planning of radar sensor network for inland waterways surveillance. In: Book Series: European Radar Conference-EuRAD, 6th European Radar Conference (EURAD 2009), pp. 501–504. Rome (2009)
7. Wawrzyniak, N., Hyla, T.: Managing depth information uncertainty in inland mobile navigation systems. In: Kryszkiewicz, M., et al. (eds.) Book series: LNAI, RSEISP 2014, vol. 8537, pp. 343–350. Springer, Heidelberg (2014)
8. Abras, C., Maloney-Krichmar, D., Preece, J.: User-centered design. Encyclopedia of Human-Computer Interaction. Sage Publications, Thousand Oaks (2004)
9. Fastenbauer, M., Sattler, M., Schilk, G.: River Information Services for commercial users in the Inland Waterway sector. In: IEEE on Logistics and Industrial Informatics, LINDI 2007, pp. 31–36, (2007)
10. Ramakrishnan, R., Baptist, A., Ercegovac, V., Hanselman, M., Kabra, N., Marathe, A., Shaft, U.: Mass collaboration: a case study. In: Proceedings of the International Database Engineering and Applications Symposium (IDEAS'04), pp. 133–146. IEEE (2004)
11. Hudson-Smith, A., Batty, M., Crooks, A., Milton, R.: Mapping for the masses: accessing web 2.0 through crowdsourcing. Soc. Sci. Comput. Rev. **27**(4), 524–538 (2009)
12. Heipke, C.: Crowdsourcing geospatial data. ISPRS J. Photogramm. Remote Sens. **65**(6), 550–557 (2010)
13. Waze Mobile. Real-time maps and traffic information based on the wisdom of the crowd [Online]. http://www.waze.com/homepage/
14. Jamsa, J., Luimula, M.: Advanced Car Navigation - Future Vehicle Instrumentation for Situation-Aware Services. In: 12th IEEE International Conference on Mobile Data Management, vol. 2, pp. 7–10. (2011)
15. Przyborski, M., Pyrchla, J.: Reliability of the navigational data. In: Klopotek, MA., Wierzchon, ST., Trojanowski, K. (eds.) International Intelligent Information Systems/Intelligent Information Processing and Web Mining Conference (IIS: IIPWM 03). Advances in Soft Computing, pp. 541–545. Zakopane (2003)
16. Lubczonek, J., Stateczny, A.: Concept of neural model of the sea bottom surface. In: Rutkowski, L., Kacprzyk, J. (eds.) Neural Networks and Soft Computing Book Series: Advances in Soft Computing, pp. 861–866. Zakopane (2003)
17. Stateczny, A.: Artificial neural networks for comparative navigation. In: Rutkowski, L., Siekmann, J., Tadeusiewicz, R., et al. (eds.) Artificial Intelligence and Soft Computing - ICAISC 2004. LNAI, vol. 3070, pp. 1187–1192. Springer, Heidelberg (2004)
18. Stateczny, A., Wlodarczyk-Sielicka, M.: Self-Organizing Artificial Neural Networks into Hydrographic Big Data Reduction Process. In: Kryszkiewicz et al. (eds.) 2014 Joint Rough Set Symposium, LNAI, pp. 335–342, Springer, Heidelberg (2014)
19. Newcomer, E.: Understanding Web Services: XML, WSDL, SOAP, and UDDI. Addison-Wesley Professional, Boston (2002)
20. Kazimierski, W., Zaniewicz G., Stateczny, A.: Verification of multiple model neural tracking filter with ship's radar. In: Kulpa, K. (ed) on 13th International Radar Symposium (IRS). Book Series: International Radar Symposium Proceedings, pp. 549–553, Warsaw (2012)

Author Index

A
Anzilli, Luca, 49
Apolinarski, Michał, 257

B
Bielecki, Wlodzimierz, 309, 409
Borgest, Nikolay, 147
Burak, Dariusz, 249
Burduk, Robert, 85

C
Cariow, Aleksandr, 423
Cariowa, Galina, 423
Chmielewski, Leszek, 227

D
Derezińska, Anna, 395
Dichenko, Sergey, 279

F
Facchinetti, Gisella, 49
Finko, Oleg, 279
Forczmański, Paweł, 157
Frejlichowski, Dariusz, 181

G
Globa, Larysa, 335, 345, 359
Gościewska, Katarzyna, 181
Gromov, Aleksey, 147
Gromov, Andrey, 147
Grosser, A., 237

H
Higami, Yoshinobu, 121
Hyla, Tomasz, 435

I
Idzikowska, Ewa, 269
Imada, Akira, 3

J
Janowicz, Maciej, 227

K
Kaleta, Joanna, 227
Kapruziak, Mariusz, 11
Karshiyev, Zaynidin, 385
Kazimierski, Witold, 435
Kholod, Ivan, 385
Klęsk, Przemysław, 11
Klimek, Andrzej, 95
Kobayashi, Shin-ya, 121
Kochegurov, Alexander, 27
Kochegurova, Elena, 27
Konys, Agnieszka, 191
Korovin, Maksim, 147
Kot, Tetiana, 345
Kovalskyi, Mykhailo, 335
Król, Tomasz, 171

© Springer International Publishing Switzerland 2015

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2

Kryvenko, Oleksandra, 359
Kubanek, Mariusz, 139
Kupina, Natalya, 27
Kurkowski, M., 237

L
Luntovskyy, Andriy, 293

M
Marczeski, Paweł, 157
Mavlonov, Kamoliddin, 121

N
Nowosielski, Adam, 157

O
Olech, Bogdan, 11
Oleksenko, O., 345
Orłowski, Arkadiusz, 227

P
Palkowski, Marek, 309, 409
Piegat, Andrzej, 63
Pietrzykowski, Marcin, 107
Piątkowski, J., 237
Pluciński, Marcin, 37
Pukhkaiev, D., 345

R
Rejer, Izabela, 95
Rogoza, Walery, 371
Rozenberg, Leonard, 171

S
Shorov, Andrey, 385
Shtogrina, Olena, 359
Smorawa, Dorota, 139
Spillner, Josef, 293
Stryzhak, Oleksandr, 335
Szymoniak, S., 237

T
Ternovoy, Maksym, 359
Tomaszewska, Karina, 63
Twardochleb, Michał, 171

V
Vasyutynskyy, Volodymyr, 293

W
Wawrzyniak, Natalia, 435
Wiliński, Antoni, 207

Z
Zabłocki, Michał, 73, 207
Zbierski, Maciej, 321